Quality control of a diagnostic tool through qualitative and quantitative measurement assessment of field testing

Martin Jidegren
Tushar Gupta

Examensarbete: LIU-IEI-TEK-A—15/02328—SE
2015-06-05
Department of Management and Engineering
Quality control of a diagnostic tool through qualitative and quantitative measurement assessment of field testing

Authors
Tushar Gupta
Martin Jidegren

Supervisor, Scania
Sophie Höglund

Supervisor, Linköping University
Bozena Poksinska

Examiner, Linköping University
Mattias Elg
Acknowledgements

The last part of the master program Industrial Engineering and Management at Linköping University has been to conduct a master thesis of 30 credits. A thesis can be conducted both practically at a case company and theoretically, where this is practical thesis that has been conducted at Scania AB in Södertälje, Sweden. We have studied several subjects related to quality and have a genuine technology interest, thus the assessment process of the quality of a diagnostic tool is something that interests us. This study has consequently been a deepening of our understanding of the field of quality.

This thesis would not have been able to conduct and finalise without the help of certain key persons. They have guided us through the entire process and help us interpret, answer and take decisions regarding questions that have arisen during the study. The people we would like to thank especially are:

**Sophie Höglund – Supervisor at Scania AB**
Sophie has, during the entire study, made sure that we have gotten the help we needed and has been the biggest contributor of information surrounding the project. Sophie has pointed us in the right direction to where we could find the information of interest and also help us conclude what to look for. To make sure we have been track, she set up a weekly meeting where the progress of the thesis were discussed.

**YSPV – Office group at Scania AB**
YSPV has been a crucial contributor of the gathering of information. They have put aside time for interviews, answering spontaneous questions and have provided general consultation.

**Bozena Poksinska - Supervisor at Linköping University**
Bozena has contributed greatly with the structure and academic writing of the report. This has been done through a series of seminars where the thesis has been reviewed and improved.

**Mattias Elg – Examiner at Linköping University**
Mattias has provided his critical comments and expertise to improve the structure and academic writing of the report.

---

Martin Jidegren

Tushar Gupta
Summary

The purpose of this study is to develop a method to qualitatively and quantitatively measure and assess the field testing of a diagnostic tool by identifying the parameters that are relevant to assess a field test. The study is conducted at Scania CV AB, Södertälje, Sweden, a world leading manufacturer of trucks, buses and industrial and marine engines, where a method to assess the field test of their diagnostic currently does not exist.

The study follows a deductive approach while taking a positivistic and hermeneutic perspective. The relevant theories and literature such as quality development and software testing are described to give a better understanding of the study. The study is conducted in four main steps- description of present situation, situation analysis, development of the assessment approach or framework and evaluation of the framework.

The empirical information gathered from numerous interviews and meetings is presented in the description of present situation along with the various data sources available. The collected data from different databases is analysed where hypotheses are formulated based on the different influencing parameters for field testing. The correlations between the parameters are then calculated and analysed to verify the hypothesis as True or False. The ECU updates are also analysed to show that the ECU updates performed during field testing is a good representation of the actual usage after release.

The framework to assess the field test is then developed using the available data and analysis made. A holistic view is taken to include the processes before and after the field test in the framework. The framework is in the form of an Excel workbook where data is either copied from databases or manually entered and relevant graphs describing the field test are generated automatically. The time period to be displayed on the graphs can be selected manually. This gives a good base to take decisions about how a field test has gone and whether or not the software is ready for release. Based on the correlation of the different parameters, a table with different key values of how much field test usage that should be conducted based on the number of implemented change requests are presented. Thus the result is that the most important attributes to consider for a field test are the amount of implemented changes where each field test usage occasion increases the chance of finding potential faults in the software of the diagnostic tool.

An unrestricted framework is also described using data that may be available, but currently difficult to utilise effectively. Thus the recommended future work is represented by this framework which describes what information that can be obtained from different data sources and how they can be used to get a detailed understanding of what exactly has been used during field testing as well as after the software has been released.

The framework is assessed in the last step and its uses along with limitations are described. The difficulty in describing the success of software testing is also discussed to give a good context to the framework and understand its utility.
Important abbreviations

YSPV – Quality, Integration and Test; Department at Scania
SDP3 – Service and Diagnostic Product 3
TR – Trouble Report
CR – Change Request
ECU – Electrical Control Unit
VCI – Vehicle Communication Interface
SOPS – Scania On board Product Specification file
## Table of Contents

1. **Introduction** ................................................................................................................. 1  
   1.1 Background .................................................................................................................. 1  
   1.2 Purpose ....................................................................................................................... 2  
   1.3 Research questions ...................................................................................................... 2  
   1.4 Case company description ......................................................................................... 2  
   1.5 Limitations and delimitations ..................................................................................... 3  

2. **Research methodology** ............................................................................................... 4  
   2.1 Our scientific research ............................................................................................... 4  
   2.2 Our method ................................................................................................................ 6  

3. **Theoretical framework** ............................................................................................... 15  
   3.1 Quality ....................................................................................................................... 15  
   3.2 Software quality ......................................................................................................... 17  
   3.3 Process and process orientation ............................................................................... 20  
   3.4 Histograms ................................................................................................................ 21  
   3.5 Pareto chart ............................................................................................................... 22  
   3.6 Scatter plots ............................................................................................................... 22  
   3.7 Performance measurement ....................................................................................... 23  
   3.8 Accuracy of estimates .............................................................................................. 24  

4. **Description of current state** ....................................................................................... 26  
   4.1 General empirical information ................................................................................... 26  
   4.2 Process mapping ........................................................................................................ 29  
   4.3 Available data ........................................................................................................... 35  

5. **Situation analysis** ........................................................................................................ 38  
   5.1 Product updates ......................................................................................................... 44  

6. **Framework development** .......................................................................................... 51  
   6.1 Implemented changes ............................................................................................... 51  
   6.2 Field testing ................................................................................................................ 52
6.3 Product updates .......................................................................................................................... 53
6.4 Restricted Framework ............................................................................................................... 54
6.5 Unrestricted framework .......................................................................................................... 56

7. Evaluation of the frameworks ..................................................................................................... 57

8. Conclusion .................................................................................................................................... 59

9. References ..................................................................................................................................... 60

10. Appendix ..................................................................................................................................... 63

Appendix A: Formal interviews ...................................................................................................... 63
Appendix B: Example of raw data from JIRA ............................................................................... 65
Appendix C: FRAS raw data ............................................................................................................ 66
Appendix D: Testlink raw data ........................................................................................................ 67
Appendix E: Cognos raw data ......................................................................................................... 68
Appendix F: Example of how the data in Qlikview is presented ................................................... 69
Appendix G: An example of the data found in SDP3 usage ........................................................... 70
Appendix H: Summary of data ........................................................................................................ 72
Appendix I: Correlation values ....................................................................................................... 73
Appendix J: Distribution of ECU failures ....................................................................................... 74
Appendix K: Restricted framework ................................................................................................ 76
Appendix L: Graphs generated in the restricted framework ............................................................ 82
List of Figures

Figure 1: Overview of our research ................................................................. 5
Figure 2: Analysis model .................................................................................. 6
Figure 3: Job type usage distribution ................................................................ 12
Figure 4: Distribution of ECU failures .............................................................. 13
Figure 5: Illustration of the quality field’s development .................................... 17
Figure 6: Process flow chart .............................................................................. 21
Figure 7: Histogram .......................................................................................... 21
Figure 8: Pareto chart ....................................................................................... 22
Figure 9: Scatter plot ......................................................................................... 23
Figure 10: YSPVs internal stakeholders and external customers .......................... 27
Figure 11: SDP3 connection order .................................................................... 28
Figure 12: SDP3 connection to vehicle .............................................................. 28
Figure 13: Overall process ................................................................................ 30
Figure 14: Inputs to the test plan ....................................................................... 30
Figure 15: Test plan process .............................................................................. 32
Figure 16: Overall test process for SDP3 development ........................................ 34
Figure 17: In-house testing per version .............................................................. 38
Figure 18: Fixed TRs and implemented CRs per version ...................................... 39
Figure 19: No. of uses during field testing per version ........................................ 40
Figure 20: No. of TRs found during field testing with corresponding severity ......... 40
Figure 21: Time duration for field testing per version .......................................... 41
Figure 22: Issues found after release per version ................................................. 42
Figure 23: Development of the product updates for field testing .......................... 44
Figure 24: Development of the product updates after release .............................. 45
Figure 25: Proportion of updates during field testing per ECU per version .......... 45
Figure 26: Proportion of updates after release per ECU per version ................... 46
Figure 27: Proportion of updates per ECU in field test and after release in 2.20 .... 46
Figure 28: Proportion of updates per ECU in field test and after release in 2.21 .... 47
Figure 29: Proportion of updates per ECU in field test and after release in 2.22 .... 47
List of Tables

Table 1: List of formal interviews .................................................................................................................. 10
Table 2: Correlation table with issue severity and no. of issues found after release ................................. 41
Table 3: Correlation values ........................................................................................................................ 42
Table 4: Hypothesis result ............................................................................................................................ 43
Table 5: ECU comparison between field testing and after release ............................................................. 50
Table 6: Data to consider when assessing the field test ............................................................................. 51
Table 7: Data collection table ....................................................................................................................... 54
Table 8: Target values .................................................................................................................................... 55
1. Introduction

This chapter presents information that gives the reader an understanding of why this study is conducted and introduces the reader to the importance of testing in order to ensure quality.

1.1 Background

The initial sales occasion is the starting point for a potentially long lasting seller-buyer relationship in the vehicle industry, where the development and orientation of this relationship is largely determined by how the customer is treated and perceives the after-sales service. After-sales service was traditionally considered to be an unavoidable cost due to breakdowns or wear and tear of products and was looked at negatively by management, but this view has changed over the years and it is now looked at as a business opportunity (Saccani, et al., 2007). After-sales service is now considered to play a vital role in gaining the customers affection and increasing customer loyalty even when a failure occurs. This argument is further strengthened by that the after-sales are acknowledged as a major contributor to an organisation’s profitability where the revenues derived from the after-sales market has been found to be four to five times larger than the new product market (Bundschuh & Dezvane, 2003). Thus the after-sales service becomes an important aspect for an organisation’s ability to survive and compete.

The importance of the after-sales service is very apparent in the truck manufacturing industry, where it includes a broad range of activities such as service centres, driver training, sales of spare parts, financing, monitoring and performance control and technical support. The main idea is to aid the customer in the efficient usage and disposal of goods. Hence the after-sales becomes crucial for both keeping customers satisfied and retaining them. This broad range of after sales services also has a significant financial impact and can produce revenues of up to four times the initial purchasing cost (Alexander, et al., 2002). This is an increasingly important and relevant aspect for a truck manufacturers’ business as vehicles today incrementally become more advanced and technology has come to play an ever more central role.

As the complexity of the technology on which todays vehicles are based increases, so does the needed level of competence and knowledge of the customers to handle and understand the vehicle. This is evident at service workshops in particular, where the increased electronic technology in vehicles has increased the complexity for mechanics to service vehicles. In order to properly be able to service and entertain trucks and buses after the initial sales point, assistance of some sort is needed. A diagnostic tool has been found to be very useful for testing internal electrical components to troubleshoot a truck or bus and find the root cause of a problem without taking the entire vehicle apart. Therefore truck and bus manufacturers have much to gain by providing diagnostic tools for troubleshooting as it both generates an additional source of income and simplifies the troubleshooting at service centres which in turn, hypothetically, leads to customer satisfaction.

In order for the mechanics to use a diagnostic tool, they have to be confident in its abilities and quality, otherwise they risk causing serious problems with vehicles which could endanger the safety of the customers. Thus a diagnostic tool must be quality assured through extensive testing to instil this confidence. While testing does not directly reduce bugs and errors or improve the quality of a diagnostic tool, it does give an idea about the present quality of the tool and also identifies errors so that they can be tackled and removed (Hambling & Morgan, 2010). Testing of the tool in-house is limited to the vehicles and configurations that are available as the testing site. Due to the large number of different electrical
configurations of vehicles and the complex nature of these electrical systems, it is not possible to test all features of the diagnostic tool on the different configurations in-house. One method to broaden the testing range is to conduct field testing, which means that service centres and customers are provided with the diagnostic to use before the actual release. Thus their usage is included as a form of quality control that improve the base on which one takes the final decision to release the diagnostic tool on time or if improvements needs to be done before release.

Field testing not only provides a wider range of vehicles to test on, but also helps improve the relationship between the workshops and the automobile manufacturer, thus bringing them closer to each other. The drawback of field testing is that it is difficult to control the flow of information and receive the right feedback from the workshops. Also, it is nearly impossible to control how the workshops use the tool, which calls the reliability of the feedback received into question. The reason for this is that the workshops view it as something additional and time consuming. It is however important to have a good method of collecting data from the field testers and analysing the received feedback in an appropriate manner to gain useful information from the field tests as well as determine the quality and reliability of the tests. This method to assess the field test can be developed through qualitative and statistical analysis of the current situation. Hence, Scania, a vehicle manufacturer who currently conducts such field testing of their diagnostic tool as a part of their quality control approach, is selected for this study. This is also relevant for Scania as their after-sales market is growing steadily and as a consequence is becoming ever more important.

1.2 Purpose
The purpose of this thesis is to develop a quality control method for a field test conducted for a diagnostic tool used during the after sales service of vehicles. The quality control method shall generate qualitative and quantitative measurements for field tests that are comparable and assessable.

1.3 Research questions
What attributes are relevant and the most important to consider when assessing the quality of a field test for a new release of a diagnostic program?
What data can be collected during a field test of a diagnostic program?
How should a method for assessing field testing be developed?

1.4 Case company description
The case company for this study is Scania CV AB. It was founded in 1891, is a world leading manufacturer of heavy trucks and buses, as well as industrial and marine engines. Scania was developed through the merger of two companies, the first of which, Vagnsfabriksaktiebolaget i Södertälje or Vabis for short, was founded in 1891 and initially manufactured railway carriages before moving to cars and trucks a few years later. The other company, Maskinfabriks-aktiebolaget Scania, was founded as a bicycle manufacturer in 1900 in Malmö and also moved to manufacturing cars and trucks a few years later. The two companies merged in 1911 and Scania-Varbis was formed. Then in 1969, Scania-Vabis merged with Saab and the new name for the company became Saab-Scania, only to be changed again in 1995 as Scania became an independent corporation. In 2008 Volkswagen became the main shareholder with 68.6 percent of the votes, which still is accurate today.
Scania has been headquartered in Södertälje, Sweden since 1912, but its operations have grown to over 100 countries worldwide with production in 9 countries in Europe, South America and Asia and consequently, they have built and delivered over 140,000 trucks and buses worldwide. Scania has been a strong player in the international market and has recorded profits every year for more than 70 years, in spite of the truck market undergoing various periods of recession during these years. In 2014, Scania had sales of over 92,000 MSEK and delivered 73,015 trucks, 6,767 buses and 8,287 engines for industrial and marine purposes. Scania has over 42,000 employees worldwide with approximately 18,400 working with sales and services, 3,500 on research and development and 12,400 at production units and regional product centres (Scania, 2015).

Scania’s vision is to be world leading in its field by creating lasting value for its customers, employees, equity owners and other interested parties. Scania’s identity as an organisation has been derived from its customers, product and services, which have resulted in three core values, “customer first”, “respect for the individual” and “quality”. It is these core values that keeps the organisation together and constitutes the base for Scania’s company culture, leadership and success.

Scania’s module product system that consists of a few major main components which enables a big range of customisation of the products and at the same time keep the product development-, production- and spare parts handling costs at a low level. This allows Scania to deliver vehicles and engines, at a low cost, based on specific customer requirement that lowers the customers overall fuel economy. This way Scania tries to maintain sustainable economic growth for the company, its customers and the society. This commitment is shown as Scania cooperates with different agencies, customers, organisations to offer sustainable, operationally reliable, energy efficient products and solution that improve the effectiveness and thus contributes to a better society.

1.5 Limitations and delimitations

Certain limitations and delimitations were made to clearly define the scope of this thesis. First, the thesis is limited to the field tests conducted for Scania’s Service and Diagnostic Program, SDP3 and will not consider tests for other software. Next, the developed model will solely measure the quality of the testing process of the field tests and not the actual results of the field tests or the quality of the software. The authors will not perform any coding of software nor develop any software tool for the purpose of the thesis. Furthermore, the knowledge of the different options in SDP3 by the mechanics can be a limitation as it determines what kind of data we will receive. This way we cannot know if the repair being made was necessary or if could been corrected in another way and we will not receive any information of how the repair was made. Due to the time frame of the thesis being limited to 20 weeks, the implementation of the method will not be conducted within the thesis.
2. Research methodology

In this chapter scientific suitable research methodologies addressed in this study are presented. It culminates in how this thesis has been conducted together with how this study can ensure to be valid, reliable and generalizable.

2.1 Our scientific research

Our research orientation is mainly positivistic as we have adopted the objectivity that it advocates where the core principle is that a specifically formulated hypothesis should be testable through observations and tests. A condition to this is that the researcher is not allowed to in any way to be personally affected or personally affect the result of the observation or test (Patel & Davidson, 2011). We have addressed each observation and data collection occasion with open minds where the aim has been to, based on logical reasoning, identify the process behaviour. This approach comes naturally as a large amount of quantitative data is collected during the field test of the diagnostic program, thus making the orientation of the research design as quantitative.

The sampling frame that has been used to limit and select the data relevant to be collected during a field test is non-probability sampling where the snowball approach to find the data has been used. Non-probability sampling is when the researcher actively is choosing the units from the population or system in question that are to be included in the study, i.e. the sampling is not randomised and thus certain units have a higher chance of being included. Snowball sampling is when the research initiate contact with a group of units that happens to be close by, e.g. workplace associates, which then is used to get in contact with more possible units and thus widen the range of the sample (Bryman, 2008). In this case it means that the field testers have been picked by Scania and that we have approached employees near us at the office to retrieve the desired information. This is done either by booking a meeting where a formal interview could be held or by an informal interview directly, where the outcome was that they either have provided us with the desired information or directed us to where we could acquire the data in question. The data is then collected based on availability and importance of certain chosen categories. Thus, a qualitative orientation in the research design is present as both unstructured and structured interviews have been used to gather data.

Both primary and secondary data is collected, where interviews with Scania employees who possess the knowledge in question are considered to generate primary data whilst the numerical data retrieved from different data bases is defined as secondary data, which is in accordance with what stated by Patel & Davidson (2011). Furthermore, we participated in and conducted direct observation of the work carried out at the company to gather raw data and documents containing relevant information for the study. This is done by joining testers at Scania in their work when testing vehicles, where we both observed and actually used the diagnostic tool to perform tests. Thus, the hermeneutic perspective also has played a significant role as certain data sets containing information, which actual meaning has been unclear, has been needed to be interpreted. These interpretations has been done based on our previous experience and perception of this kind of data, thus a subjective perspective has been added to the study and the overall research perspective is both positivistic and hermeneutic according to the definitions given by Patel & Davidson (2011).

The study’s research approach is deduction where existing theoretical concepts are gathered, from which the base of the study is relied on to determine what attributes are the most important to assess a field
To find relevant theories, a search for literature is conducted, where different approved books and scientific articles are identified and used to gain sufficient knowledge to address the research questions. Based on these theoretical concepts, an analysis model is developed that is customised towards the purpose of this study. Thus this research is classified as a case study since the collected theory is tested towards a specific case.

The reliability of the study is strengthened as all the collected data and measurements have been continuously evaluated as the same set of information has been collected and measured several times to detect deviations in the data. To further ensure the reliability, knowledgeable people at the company and supervisors at Linköping University, have been reviewing the way the information has been collect and processed as well as if the right information has been collected. Thus the validity of the research also is authenticated and further strengthened as we have triangulated different sources of information to confirm and justify our statements about the phenomenon.

The generalizability of the study is limited to the scenario of only having a restricted amount of data available regarding the particular point of interest, where one based on this data wants to make a future prediction based on statistical analysis. Thus the framework of the analysis model can act as a base for other studies where the four general steps is universally applicable. See Figure 1 below for a visualisation of our chosen research methodology with its corresponding truth criteria.

Figure 1: Overview of our research
2.2 Our method

In this section our method is presented. It contains a general description of our approach and the general steps performed in this study to solve the presented problem, which can be seen in the analysis model in the section below. The analysis model is then further followed by a more detailed description of how the study has been conducted.

2.2.1 Analysis model

The scientific approach of this study is deduction, which concept is to collect empirical information that is then analysed based on existing theories. This way one can strengthen the objectiveness of the research as it is not based on the researchers own values and perceptions but only on existing and proven theoretical concepts (Befring, 1992). Thus an analysis model is created that is derived from the principles of the SAMIE model, which is named after its five main components- Select, Analyse, Measure, Improve and Evaluate. It is an adaptable model for continuous process improvement described by Chang (2000), and has been adapted to suit this study’s purpose, where the general approach of how to improve an existing process is focused on. This analysis model has been created since no general procedure for generating an assessment framework for field testing exists. The model enables a systematic and structured way of collecting empirical information, which constitutes the base of a good analysis. As a proper analysis is made of relevant empirical information, reliable and valid result can be generated. Then as the foundation of this analysis model is built around a continuous improvement approach, it is reusable and can be looped for further utilisation by the case company after this study is finalised.

The model is adjustable and consists of four main steps, where the different tasks to perform for each step can be adjusted to better suit the case in question. This analysis model is consequently the tool that enables the answering of the presented research questions and thus achieving this study’s purpose. In Figure 2 below, a visualisation of the analysis model adapted for this study is presented.

<table>
<thead>
<tr>
<th>1. Description of present situation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Collect general empirical information of the processes connected to field testing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. Situation analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Analyse field testing data</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. Development of field test assessment approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Develop a framework to assess the field test without restrictions</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Evaluation of the generated frameworks</td>
</tr>
</tbody>
</table>

Figure 2: Analysis model
1. Description of present situation

Here all the empirical information that is relevant to comprehend the study is presented. The description of the present situation is divided into the three focus areas seen below.

1.1 Collect general empirical information

This is the first step to conduct in order to get an overall picture of the operations carried out that are connected to field testing. The different groups and the activities that they are carrying out are identified and described in the chapter Description of current state.

1.2 Process mapping

In this step, the field testing process is mapped through a flowchart and all the relevant steps along with their connections to each other are described. An explanation of what each step includes is also presented to clarify what happens in the different steps of the process. Then the information flow between these steps is also identified and explained. These process maps have been controlled and confirmed to be correct by our supervisor at Scania. The process mappings are presented in the chapter Description of current state.

1.3 Identify available data

In this step, the available qualitative and quantitative data that can be and is being used right now to assess the field test is identified. Suitable time periods and other relevant parameters, such as the version numbers for which the data set should be collected, are determined and the selected data is collected. This is collected and identified through formal interviews and discussions with employees at Scania. Then, in collaboration with our supervisor and the test leader at the department, the data being used and the data that would be preferable to use are identified. Then, an explanation of what this data actually is and what information it corresponds to is described along with the source and form of the data to simplify the future handling of it. Furthermore, the possible qualitative and quantitative measurements are identified, which all are conducted in collaboration with our supervisor and the test leader.

2. Situation analysis

Here the empirical information is analysed and the information that can be retrieved and inferred from the collected data is presented. The value of the data is assessed together with its usability, where the measurements and their accuracy is determined. The analysis is divided into the two focus areas seen below.

2.1 Analyse field test data

In this step all the data retrieved from the field tests is analysed to see what information it actually contains and how it can be best used. The accessibility of the information contained within the data is also assessed and it is determined whether or not it is relevant to use. If it is deemed as not usable without some sort of external processing, the necessary action required is evaluated in terms of the possibility and ease of realisation together with its potential value addition. Then, data from different periods of time is assessed and its value is determined.
2.2 Analyse other available data

In this step all the other data that is not retrieved from field testing that still is relevant to consider is analysed in order to assess and understand what the field testing data actually means. Here internal data from Scania is assessed and as mentioned above the value of the data corresponding to different time periods is determined.

3. Development of field test assessment approach

Here different frameworks for assessing the field test are generated by brainstorming based on the analysis of the collected empirical information and relevant software quality metrics. Two different approaches of how to assess the field test are presented, one with and one without restrictions. This is done to both provide Scania with a functioning framework that they can start using immediately and to give them suggestions of future work to conduct in order to improve the framework.

3.1 Develop a framework to access the field test without restriction

Here a framework is developed without any restrictions regarding the data and information that should be collected together with how it should be assessed i.e. how the assessment of the field test ideally should be done if all data and information that is relevant and desirable is available. An example could be that it is necessary to create a program to retrieve a particular set of data that otherwise cannot be utilised, which might not be plausible in reality and one therefore needs to be content without that data when adopting a realistic perspective.

3.2 Develop a framework to access the field test with restriction

Here a framework of how and what data that should be collected is developed in regard to the conditions and restrictions that are present. This is the development of the framework that is possible to implement and utilise immediately. This framework is not dependent on any external work in order to function.

4. Evaluation

This part consists of a discussion that culminates into conclusions and recommendations regarding which framework that is the most suitable to use depending on different scenarios. This is conducted in collaboration with our supervisor and the test leader at Scania, where the actual use and implementation value of the frameworks are discussed and decided upon.

4.1 Evaluation of the generated frameworks

In this step the different frameworks that have been generated are evaluated, where the different pros and cons of the respective framework is discussed and elaborated thoroughly. Based on the comparison and evaluation of these frameworks, a recommended framework is presented together with the future work that should be conducted in order to enhance the assessment possibility of field tests.

2.2.2 Data collection

There are two different research designs of a study, which are quantitative and qualitative. The quantitative approach implies that the research has a predetermined structure that simplifies the management of gathered data. This research strategy is based on quantified data collection where the deductive approach describes the researcher’s relation between theory and practice (Bryman, 2008).
qualitative research methodology is distinctive in regards to that there is not necessarily a limit on the number of influencing factors and variables. Qualitative studies tend to generate a more detailed description of the research topic than the quantitative approach. Thus the meaning of qualitative research is to detect and distinguish something specific in the information being investigated to find meaning and significance (Bengtsson, et al., 1998).

In this study both the quantitative and the qualitative research designs are applied to enable large quantities of data to be collected where a profound insight in the information still is achievable. Thus the study is initiated with interviews and unstructured talks with the employees at the office to get an overview and first understanding of the present situation, which in turn reveals what information is relevant to collect and where it can be retrieved from. Consequently the initial data collection has a qualitative approach where interviews, meetings and unstructured talks are performed. To further increase the understanding and to confirm or reject the qualitative data, raw data is retrieved, which corresponds to the quantitative aspect of the study. The quantitative approach also has significant importance as it enables unbiased results to be generated, based completely on the analyses of raw data without the influence of personal perceptions. The raw data is collected through databases and different documents as described later in the chapter Description of present state.

This consequently means that both primary and secondary data are collected in order to get a holistic perspective of the situation. Primary data is the data obtained directly from the original source where interviews and observations are commonly used methods to obtain the empirical data. Secondary data is information obtained indirectly from the main source, i.e. from someone or something else than the original source. Examples of secondary sources could be databases, literature, statistics and documents (Patel & Davidson, 2011).

The data collected in this study pertains to the various versions of the diagnostic tool. The versions for which the data is to be collected is determined through snowball sampling based on the availability of reliable data for the different versions as well as the ease of collecting the data.

2.2.2.1 Interviews
There are two main interview forms, which are structured and unstructured interviews. Unstructured interviews’ focus is on an area rather than to use a set of standardised questions thought through before the interview. This allows the interview to develop in a spontaneous way. This interview technique is advantageous when one desire to establish a broader picture of the research field in question as the interviewer is allowed to adapt to the specific situation at the time (Trochim, 2006). The structured interview approach is recognised as having a set of predetermined standardised questions that are organised in a structural manner. This simplifies the analysis of the respondent’s answers as they become easier to compare in contrast to an unstructured interview (Bryman, 2008).

To ensure that a wide holistic picture of the area can be achieved in a time efficient way, a combination of unstructured and structured interviews are held, i.e. semi-structured interviews. This way it can be assured that both the broader holistic picture and the information regarding specific cases or process are retrieved. In Table 1 one can see the formal interviews that have been conducted to collect information regarding the description of present state. To strengthen the reliability of the response, several people such as the test leader, testers and field test coordinator are interviewed to get their specific view of the process and thus be able to distinguish biases. The interviews are structured in such a way where the
interviewees are first asked questions regarding his or her work and then how that work is connected to the field testing, thus creating a natural flow and direction of the interview. To make sure that our own limited understanding of the area does not becomes a restriction, the questions are open oriented where a broad response initially is accepted to then be narrowed down as the interview proceeds. Then to steer and narrow the interview questions in a structured manner makes it is simpler to analyse the response as each reply for a question can be compared to the reply for the same question from the other respondent without putting significant effort into it (Bryman, 2008).

The interviews contain a large number of questions and when there are complex questions, the interview is divided into different categories in order to simplify for the respondent by asking all questions related to each other for a particular area at the same time. The questions are also being repeated to ensure that the interviewee understands them correctly and thus can give an accurate response. To further strengthen the reliability of the response, only employees with at least one year of experience at their current position are interviewed to ensure that they possess sufficient knowledge. In Appendix A one can see the interview form used to collect overall information that is connected to the field testing to get an holistic understanding of the process and also an interview with questions oriented towards a particular focus.

Table 1: List of formal interviews

<table>
<thead>
<tr>
<th>Employees</th>
<th>Subject</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test leader</td>
<td>Testing process</td>
<td>Scania</td>
<td>20 February 2015</td>
</tr>
<tr>
<td>Tester</td>
<td>Testing process</td>
<td>Scania</td>
<td>20 February 2015</td>
</tr>
<tr>
<td>Testers</td>
<td>ECU updates</td>
<td>Scania</td>
<td>17 March 2015</td>
</tr>
<tr>
<td>Field test coordinator</td>
<td>Various topics such as testing process, field testing, ECU updates, data availability</td>
<td>Scania</td>
<td>Continuous weekly meetings and interviews</td>
</tr>
</tbody>
</table>

The analysis of the interviews were conducted by comparing the answers between the respondents to identify dissimilarities. Where different responses were found, we brought that up and discussed it with the respondents in an informal way to find an answer representing reality. The validity of these answers and the ones where the same response were given, have been controlled with both our supervisor and the interviewees. Regarding the raw data collection from data bases, the actual meaning of the information retrieved has been discussed with our supervisor and other employees with knowledge regarding the data source in question. In addition to this, all major conclusions have been discussed and verified by our supervisor through the weekly meetings that have been taking place.

The validity of the interview questions are controlled by letting people with significant knowledge of the area give their opinion. This is also done during the interviews by encouraging and asking the interviewee for his or her opinion on the question, whether or not they are relevant and if something important has been excluded. This clarifies the focus and direction of the study and makes it easier to get a clear picture. This consequently enables more specific questions to be asked and thus information on a deeper level to be collected.

Additional informal interviews are held, where questions to which a quick and relatively easy answer is deemed possible are asked directly as and when they arise. When the person possessing the desired information in question cannot give an answer at the time for a question, a meeting or future interview is scheduled. This approach is time efficient and enables the work to have a more continuous flow as...
shortcoming in the information are not stopping the progression to the same extent as if a scheduled meeting or interview are to be done for every question or uncertainty that requires clarification.

2.2.2.2 Participant- and direct observations

Participant observation is characterised as an extremely time demanding approach, where the time span starts at a couple of months to several years. The reason for this is to familiarise the researcher with subject or area of observation thorough enough to that he or she can be counted as an experience individual in the actual field being investigated. This is valuable as it enable and ensure that the researcher can make natural observations of the phenomena in question from a knowledgeable persons point of view and thus generate information that conforms with reality (Trochim, 2006). This qualitative approach has its base in striving to conduct the observation of the subject in question as discretely as possible. The reason for this is to minimise the interference and thus the contamination of the researcher’s observation in itself. The difference between direct- and participant observation could be said to be that the observer does not try to become a part of the system being investigated but to only visually observe the process taking place (Trochim, 2006).

Participant- and direct observations are conducted to further increase the understanding of the processes regarding and surrounding field testing. The participant observations are for example joining and performing in-house testing of a vehicle and thus valuable insight in how the usage of the field testing software can be achieved. We participated in a “Testfest” where around 20 people sat down in a room and conducted actual testing on demo files, that mirrors the properties of actual vehicles. Here we found minor faults in the software of the diagnostic tool and attained real use experience. The faults found were created into issues that then later were corrected. Then, to complement and detect the biases of the software testing due to our participation, direct observations are conducted. This was done with the test leader, who showed us how he and the other testers in general are performing a test with the diagnostic tool in a truck, making it a direct observation. This was also done by joining a tester conducting tests with the diagnostic tool on a bus. However, the tests on the bus did not work out as intended and had to be cancelled during the session. Thus the entire testing process on a bus was not observed. No real biases of how the testing is conducted were identified beside that the experience testers could test what to be tested in a faster and more comprehensible way. Thus an understanding through own experience and observations of experts in action is attained.

2.2.3 Data analysis

According to Bergman & Klefsjö (2006) data, along with its analysis is the foundation for improvement projects. Also, one of the core principles of quality management is to take decisions based on facts and data (Tort-Martorell, Grima, & Marco, 2011). Data analysis refers to understanding and transforming raw, unintelligible data into useful information. In situations where there is a lot of data available regarding the point of interest, one should initiate the problem solving by analysing this data. Four approaches used to treat the data can be seen in the paragraphs below.

Review the information in a holistic way

The information is analysed holistically to get an overall view and an initial understanding of the data with the help of a graphical analysis of the information. Creating Pareto charts, scatter plots and histograms enable patterns in the data to be discovered. For example, the most commonly occurring ECU update failure can be identified. This way the possible flaws are revealed and the distribution of the data is also
obtained. An example of an identified flaw was the usage of a particular function in the diagnostic tool that emerged due to that automation testing through prepared demo-files. An example of this can be seen in Figure 3 below, where the names and numbers are removed due to confidentiality. This data could then be disregarded when considering the overall usage of the diagnostic tool. Histograms and scatter plots are great tools that are used to visualise the dispersion of the data in question and thus obtain first indication of what type of data one is dealing with (Sörqvist, 2004).

Figure 3: Job type usage distribution

Analyse the variation in the process3

The variation between the data over time is analysed statistically to further discover patterns and trends that are inherent in the process. Discovery of trends enables the possibility of predicting future expected values and performance. The distribution of the ECU failures can be seen in Figure 4 below, where the actual numbers are censored due to confidentiality. Then process variations are analysed and interrelationships determined through hypothesis testing and correlation analysis which is presented in in the chapter Situation analysis. Here it is important to understand that correlation between different parameters does not imply that are connected, but only that they seem to change in a predictable manner when considering one of the parameters (Sörqvist, 2004).
Determine goals and desired capability

To properly be able to assess and interpret the situation of interest one must determine what demands and requirements the process should be able to handle. This is either based on known technical prerequisites, on external references or a combination of both. An example of an external reference is benchmarking with other comparable processes to determine a reasonable capability for the process in question. An example here is the comparison of the ECU failures found after release between different versions of the software in the diagnostic tool, even though it does not come from an external source but an internal it can be used for comparison. Here a minimum usage determines whether or not the data can be used in order to increase the reliability of the data. This minimum usage value is determined as 200 to minimise the implication of single data points. Sörqvist (2004) states that this approach of comparing also can be used to identify what “world class” means and based on this challenging and future goals can be set. Here the goals are identified through a combination of technical expertise and statistical analysis. By analysing the ingoing parameters of the software that influences the quality of it, a minimum number of usage during field testing can be derived that corresponds to a particular number of problems found after it has been released. Thus a target number can be developed which is to be derived dependent on the correlation and connection that the different parameters display. These parameters and the correlation of them can be seen in the chapter Situation analysis.

Identify causes

In order to improve the process one must increase the understanding of it. This enables the identification of the cause of why a process has a certain capability which in turn allows for it to be influenced and consequently improved. The root cause of the identified interrelationships are investigated and determined through thorough analysis and then validated by technical experts and knowledgeable employees at Scania. This has mostly been confirmed through informal meetings and questions with the employees at the office.

Figure 4: Distribution of ECU failures
2.2.3.1 Brainstorming

According to Osborn (cited in Brown & Paulus, 1996), brainstorming is a common approach used for the generation of new, creative ideas. Brainstorming aims to generate a high number of ideas which are compared and analysed subsequently to achieve the best solution. It can be conducted individually or in groups, though research suggests that group brainstorming usually does not perform as well as all the individuals of the group brainstorming individually (Brown & Paulus, 1996). Brainstorming in a group does have some advantages however, if performed under the right conditions, and can stimulate different minds to generate a higher number of ideas with better quality. A combination of both methods can generate a higher number of ideas than either method performed alone, by first brainstorming in a group and then brainstorming alone (Paulus & Nijstad, 2003). Having a cross functional group with different competences and views produces better results in terms of number of ideas, which could lead to a better overall solution.

Brainstorming is conducted both individually and together as a group where the individual brainstorming is done first to then be merged in brainstorming sessions together. This way no major ideas are lost due to the influence of the others thinking, and all ideas deemed as relevant to someone will reach the surface. This way a broad and general approach towards what the data can be used for and how it should be used can be generated. However, the brainstorming is also generating ideas perceptions regarding the task that are incorrect and thus the brainstorming needs to be re-evaluated and conducted during the progress of the study to ensure that the generated ideas are deducted from the latest and most relevant information.
3. Theoretical framework

In order to develop an assessment method for field testing there are several theoretical concepts which one needs to be familiar with. The first concept and the foundation of which the method is to be based upon is quality. It is thus crucial to have a good understanding of what it really means, so that one can assess and determine the quality of the field test properly. It is then important to deepen the understanding of quality with its implications for a software and consequently a diagnostic tool. This way the work of improving the quality of the product from both the customer point of view, i.e. field testers, and Scania’s point of view can be done based on a solid foundation. Then to ensure quality of a diagnostic tool one must know how to conduct testing of a software in such a way that minimises the risk of crucial and quantitative failures emerges for the end users.

To fully understand what the field testing is and how it works with all its process steps, one must conduct a process mapping. This way one can identify all the activities of importance that are carried out and belonging activities influencing the main process of interest. This way an understanding can be created of how the process and its activities should and can be treated. It is important to understand how the diagnostic tool is performing and how to measure that performance in order to determine what and how to improve the product. Therefore several tools of how to handle raw data is explained, where histograms and Pareto charts are examples. These tools are good ways of visualising the data which simplifies the interpretation and analysis of it, which enables one to assess the diagnostic tool and field testing based on its fundamental raw data components. This chapter ends with an explanation of how software estimates can be done. This is an important part of the study as the data sources and the data they contain can consist of data that are incomplete and uncertain which meaning needs to be interpreted, where an estimation of its real implications must be done.

One important but yet missing theoretical concept is field testing. This is due to that no sufficient literature or scientific material regarding field testing of a software or diagnostic tool exists to the best of our knowledge. However, with these theoretical concepts, which are presented below, one is provided with a good base on which a framework of how to assess the field testing of a diagnostic tool can be developed.

3.1 Quality

The word quality comes from the Latin word “qualitas” which means “of what” or “nature of something” and is said to have been used the first time by the ancient politician Cicero during 106-43 BC. Since then the word quality has been widely used where an example is when describing different types of steel by labelling them with a certain steel quality. Over the years, several definitions of quality has been created, where a common but all too narrow definition is “conformance to requirements” (Crosby, cited in Bergman & Klefsjö, 2007, p.23). There are two different aspects of quality, one objective and one subjective. The objective aspect corresponds to the measurable point of quality, e.g. the force a steel beam is capable of handling. This is important from a production perspective as common reference points are crucial when decisions regarding development and manufacturing of products and services are to be made. The subjective aspect corresponds to the customer’s perception of the product or service. This is considered the most important aspect of quality as it is the customers experience and perception of the product or service that ultimately decides it succeed (Bergman & Klefsjö, 2007). A good definition of quality for this study needs to be wide enough to really incorporate the true meaning of it quality, and thus Bergman and Klefsjö (2007) definition of quality is suitable; “the quality of a product is its ability to
satisfy, and preferably exceed, the needs and expectations of the customers”, where product refers to either a product or a service.

In general people claim that they know what quality is and that they understand the concept of it, but when asked to specify the attributes of quality it becomes clear that there is much confusion and myth surrounding it (Dale, 2003). Even though there is no single agreed upon definition of quality in the business world today, it can be said to be used for distinguishing something, e.g. one organisation, event, service, product, etc. from another.

Quality received an increased interest in the late 1980s as international competition became more intense and customer expectations became greater (Samson & Daft, 2005). As a consequence, quality and quality improvements were identified as one of the most important aspects regarding competitiveness and companies started to incorporate quality as a cornerstone in their corporate strategies (Belohiav, 1993). Thus it became clear that quality is of strategic importance and that organisations must exploit it in order to enhance their position in the marketplace (Gadenne & Sharma, 2008).

The early 1990s started a new era in quality management where total quality management (TQM) became a popular quality orientation and the effectiveness and flexibility of the business as a whole became central. The management’s responsibility became to plan and coordinate the company’s total quality activities against set goals and objectives (Bergman & Klefsjö, 2006). To successfully do this, different concepts and methods, such as PDCA-cycle and total quality control, were used and integrated into the company’s information- and material flow at both the external and internal level. This led to a new understanding of the concept quality, where the involvement of everyone’s daily commitment became central for TQM and consequently quality.

To survive the ever changing environment in today’s increasingly aggressive markets, the old and narrow definition of quality as the reliability of a product or service is no longer a competitive factor strong enough to grant a significant advantage over other providers. It has become an expected requirement, to which companies can cope with through continuous improvements of their business and thus providing superior products or services. Montgomery (2009) defines continuous improvements as “the reduction of variability in processes and products”. Dale (2003) states that a customer focus that runs throughout the entire organisation’s activities that emphasises quality and flexibility is the main mean to use for companies to handle competitive threats. This has led to both small and large companies in both the manufacturing and service sector to attempt to evolve their quality approaches with the help of industrial and academic leaders in the field of quality management (Anderson, et al., 1994).

To further improve and keep customers satisfied it is important to be able to ensure a certain quality throughout the entire life cycle of a product or service that an organisation offer. Therefore it is common that organisations issue a quality assurance policy, which is incorporated from the inception of an idea until it ceases to exist. This way companies can communicate an assurance towards the customers that their product or service will endure. The main objective of a quality assurance policy is to create confidence that the product or service will perform as expected and meet the customer requirements. To achieve this, the emphasis should be on pursuing corrective and preventive actions throughout a process where non-conformance investigations and procedures are carried out in a thorough manner to assure quality at each stage of the process. This way a reliable and predictable outcome can be created (Dale,
This creates an advantageous differentiation between organisations applying the concept of quality assurance and those that are not, improving the former ones’ competitiveness.

### 3.1.1 Quality development

The quality development of the last decades can be said to have moved towards the initial product development phase, whereas before quality work was governed by inspection. One had inspections of finished products and removed defect products for reprocessing. This kind of defensive quality approach has diminished significantly and is called quality inspection. It was quality control that took over where one tries to detect defect units during the production process and to make required adjustments to the process itself in order to prevent defect units from being produced. The next step in the quality development was to prevent defect units from being produced by creating favourable conditions before starting the production. The focus was to create suitable routines for how to handle incoming customer complaints, measurement, equipment, material and how to allocate responsibility. This can be summarised as a quality system, where these sorts or activities is called quality assurance. Quality assurance is however not to be mistaken for assuring customer satisfaction, but only the intended quality of what is being produced (Bergman & Klefsjö, 2007).

This kind of development has continued in the field of quality, where the focus lies on creating favourable conditions from the start to prevent defects from occurring. The thinking of systematically gathering information regarding customer demands and wishes and by conducting planned experiments to create robust processes one can prevent defect and non-profitable products reaching the market. Quality management includes the quality inspection, quality control and quality assurance approaches and is accounted as an integrated part of an organisation, which purpose is to continuously work with improvements (Bergman & Klefsjö, 2007). See Figure 5 for an illustration of above mentioned terms and an overview of the quality development.

![Figure 5: Illustration of the quality field’s development (Bergman & Klefsjö, 2006).](image_url)

### 3.2 Software quality

The Institute of Electrical and Electronics Engineers (IEEE) define software as “computer programs, procedures and possible associated documentation and data pertaining to the operating of a computer
system” while software quality is “the degree to which a system, component or process meets specified requirements” or “customer or user needs or expectations” (cited in Galin, 2004, pp. 15, 24). A software program can fail to meet the specified requirements or customer expectations due to a variety of reasons such as incorrect or missing requirements, unrealistic requirements, system faults, program faults and incorrect code (Pfleeger, 1998).

3.2.1 Software testing

To ensure that the customer does not face inconveniences due to these errors, all software must be tested before being released for customer use. Myers, et al. (2012, p. 6) define software testing as “the process of executing a program with the intent of finding errors”. According to them, testing is a destructive process and by treating it as such, one can identify errors in the program which would enable the process of getting rid of the errors.

The process of testing does not involve fixing any errors, but it provides the first step towards just that, because errors cannot be fixed unless they have been identified. Errors are mistakes made during the coding of software and are also known as bugs (Jorgensen, 2008). An error causes a fault in the software which could lead to a failure. Depending on the software type and environment, a failure could lead to the damage of companies, environment and people (Hambling & Morgan, 2010). Thus, the testing of software to find errors so that they can be solved is vital.

Testing is not only useful in identifying errors, but it also gives an indication of the quality of a software and allows the measurement of some aspects of the software quality (Hambling & Morgan, 2010). Myers, et al. (2012) agree that providing confidence in the software is the eventual goal of testing, which is done by fixing errors that have been discovered and determining the quality of the software. Different metrics are used to get an indication of the quality of the software during testing such as time since the identification of a severe issue, current number of severe issues, bug find rates etc. According to Galin (2004), another indirect objective of software testing is to prevent errors in the future through corrective and preventive actions based on the currently known errors.

Testing is usually conducted in numerous stages, with each component of the program tested first individually to verify that they work as expected. Next, the interaction between the components is tested to ensure that work together as desired. The function of the program is tested next to verify that it has the system functionality as described by the specifications i.e. it correctly performs the tasks as intended. This is then followed by the performance test which ensures that the software works in the working environments described in the specifications. The customer then conducts an acceptance test to verify that the system functions as per their requirements, which may have been misinterpreted by the developers. If the previous tests were not conducted in the actual working environment, an installation test is performed to make sure no errors arise on site (Pfleeger, 1998). When newer versions or releases of an existing software are tested, certain tests called regression tests are conducted. A regression test ensures that changes in the current version did not insert new faults in the software. It tests parts of the software that were functioning properly in the older version and make sure that they are still functioning as expected.

The testing process is handled through test cases which are a sequence of tests, usually associated with a particular function or area in the software. Hambling & Morgan (2010) define a test case as “a set of input values, execution preconditions, expected results and execution post conditions, developed for a
particular objective or test condition, such as to exercise a particular program path or to verify compliance with a specific requirement”. Test cases put together to form a test suite which are a part of the overall test plan. While a test plan should aim to cover all parts of the software, Hutcheson (2003) argues that the purpose of testing should not be to identify and eliminate 100% of the bugs, but to eliminate the severe bugs. This is sufficient as the customer is willing to work around smaller bugs that can be dealt with easily. This view is shared by Hambling & Morgan (2010) who say that testing all possible combinations and scenarios is not possible and one should prioritise areas based on risks while testing.

3.2.1.1 Risk based Testing
One method of narrowing down the testing criteria and focus is risk based testing. According to Hambling & Samaroo (2009), risk based testing is the preparing of a test plan by prioritising tests based on level of risk. A simple risk based testing model takes the probability of failure and its likely consequence into account (Amland, 2000). This includes a detailed risk analysis to identify the risks and prioritise them. Each risk is then looked at and tests are designed to tackle them. Risk based testing ensures that the most important issues are tackled first, which gives more time to tackle them without the pressure of an upcoming deadline. In case of limited time frames for testing, it also ensures that the cases left untested are relatively less important with a lower risk of failure and impact. By measuring the progress of the test plan using suitable metrics such as remaining risks above a certain priority level, correct risks above a minimum priority level, etc., it is possible to get a view of the status of testing and remaining risk. This can enable well informed and accurate decisions to be taken about a release as the release date approaches. Risk based testing not only saves time and money, but also facilitates the development of a practical test system that can be used for future releases of the software (Amland, 2000).

3.2.1.2 Software quality metrics
According to IEEE (cited in Galin, 2004, p. 413), a software quality metric is “a function whose inputs are software data and whose output is a single numerical value that can be interpreted as the degree to which the software possesses a given quality attribute”. Software quality metrics can be used to give an indication about the quality of the software, as well as the quality of the testing process. Galin (2004) further classifies software process quality metrics into three types: error density metrics, error severity metrics and error removal effectiveness metrics.

Error density metrics measure the ratio of quantity of errors to volume of software. The quantity of errors can either be the number of errors found or the weighted number of errors depending on severity or other factors, whereas the volume of software is usually the number of lines of code or the resources required to develop the program, which is known as function points. A general formula for error density is shown below.

\[
\text{Error density} = \frac{\text{Quantity of errors}}{\text{Volume of software}}
\]

Error severity metrics give an indication of the overall severity level of the errors found. They measure the ratio of weighted number of errors to the total number of errors. The weighted number of errors is calculated based on the severity of each error found. A general formula for error severity is shown below.

\[
\text{Error severity} = \frac{\text{Weighted number of errors}}{\text{Number of errors}}
\]

Error removal effectiveness metrics give an indication of the quality of the testing by comparing the quantity of errors found during testing with the errors detected during regular operation. They measure
the ratio of quantity of errors during the development phase to the summation of the errors detected during the development phase and the errors detected during a specified period of operation. The quantity of errors can be either weighted or simple number of errors. A general formula for error removal effectiveness is shown below.

\[
\text{Error removal effectiveness} = \frac{\text{Errors during development}}{\text{Errors during development} + \text{errors during regular operation}}
\]

Another method to determine the quality of a test is fault seeding. Fault seeding is the process of inserting a certain number of faults called seeds into a program before testing and then measuring the quality of the test based on the number of seeds that were found.

3.3 Process and process orientation

Bergman & Klefsjö (2006, p. 426) define a process as a “network of activities that are repeated in time, whose objective is to create value to external or internal customers” while Harrington (1991, p. 9) defines a process as “any activity or group of activities that takes an input, adds value to it, and provides an output to an internal or external customer.” Both these definitions of a process are applicable to both, the manufacturing and service sectors. For example, in the manufacturing sector, each work station performs a task on the product, thus adding value, and the task is usually repeated on subsequent products. The product is then delivered to the next station which is the internal customer. In the service sector, the variables may be less tangible, but the basic characteristics of a process remain the same.

Kohlbacher & Reijers (2013) describe process orientation as focusing on the process involved rather than the hierarchical structure of organisations. By focussing on processes, it is possible to measure process performance and get a clear idea of the resources required for the process which could lead to the identification of possible complications that exist currently, or may arise in the future.

According to Bergman & Klefsjö (2006), processes are of three types: main or core processes, supporting processes and management processes. Main processes are those processes that directly fulfil the needs of the external customer. Support processes assist the main processes by providing the necessary resources required for the core processes. Management processes are those that take decisions regarding the goals and various other processes of the organisation.

3.3.1 Process mapping

According to Bergman & Klefsjö (2006), a suitable method to understand a process is to map the process by identifying and illustrating the different activities involved in a process in a flowchart. A flowchart uses standardised symbols to give a pictorial view of the steps of a process to obtain detailed knowledge about the process (Dale, 2003). This helps easy understanding of complex processes and can help identify non value adding activities present in the process. Mapping a process also shows how a process actually works, rather than how it is supposed to work as specified by the organisation (Brook, 2010). Flowcharts were initially used to describe production processes, but now, they can be used to illustrate all types of processes. Different symbols are used for different activities, with the common activities being start, stop, process and decision, with arrows describing the linkages between the activities. An example flow chart is shown in Figure 6.
3.4 Histograms

A histogram is used to graphically represent large amounts of data by dividing the data range into different classes or bins (Bergman & Klefsjö, 2006). A simple histogram with example data is shown in Figure 7. The number of errors recorded from different machines in a day is divided into nine bins which is displayed on the x-axis and the frequency of each class is plotted on the y-axis. The width of each bin should preferably be the same to enhance the readability and make the histogram easier to interpret (Montgomery, 2009).
3.5 Pareto chart
A Pareto chart, named after Vilfredo Pareto who was an Italian statistician, is a management tool which helps to identify the most frequent defect categories of a given data set (Brook, 2010). The most frequent categories are often the most important and thus a Pareto chart can help identify the most important categories quickly. According to Montgomery (2009), it is a graphical representation of the frequency distribution of categories of attribute data displayed in order of their frequency. In simple terms, it can be described as a bar chart which is sorted in descending order of frequency. A Pareto chart often includes a cumulative frequency line on the secondary axis which can be seen in Figure 8.

![Pareto Chart](image)

**Figure 8: Pareto chart**

It can be clearly seen from Figure 8 that categories F, C and G are responsible for over 80% of the defects and generally one should concentrate improvement activities on these categories first before moving on to the next biggest problem. This shows Juran’s (cited in Bergman & Klefsjö, 2006, p. 224) principle of “the vital few and useful many” which says that few problems, called the vital few, are responsible for a large number of defects or costs. This is also similar to the 80/20 rule which states that 80% of defects are caused by 20% of the problems. However, it should be kept in mind that the severity of some problems may be higher, even if their frequency is low.

3.6 Scatter plots
A scatter plot or scatter diagram is used to plot two variables against each other to identify relationships between them. The various factors influencing a quality characteristic can be compared with each other and the effect of each of them have on the quality characteristic as well as on each other can be determined. When plotting one variable (x) against another (y), an indication of the relationship between the variables can be inferred from the shape of the plot (Montgomery, 2009). Figure 9 shows a scatter
plot where \( x \) is positively correlated with \( y \) and tends to increase as the value of \( y \) increases. It should be noted though that “correlation does not necessarily imply causation” (Montgomery, 2009, p. 205) and the cause of some relationships may be a different unrelated parameter.

Figure 9: Scatter plot

### 3.7 Performance measurement

Measuring performance is a fundamental step in understanding and improving performance and the process behind it. According to Tonchia & Quagini (2010), performance cannot be managed or improved if it cannot be measured. Measuring performance gives a clear understanding about the current situation, providing information about the different aspects that define the performance such as production output, customer satisfaction, financial information, utilization of service, time taken etc. This information about the current situation provides clarity of the deviation from the expected performance and helps visualise how to achieve the targeted performance. Performance measurement also helps identify when the target performance has been reached, giving an opportunity to revise the target and strive for continuous improvement.

Performance is measured and reported by terms or indicators called performance measures. These measures are also used to improve performance and can be classified as key result indicators, key performance indicators, result indicators or performance indicators (Parmenter, 2010). The performance of the core activities of a business are referred to as the operational performance of the business. Tonchia & Quagini (2010) divide operational performance into external and internal, i.e. performance that is visible to customers and performance that is not visible to customers respectively. External performance is closely related to customer satisfaction, while internal performance is related to efficiency given by categories such as time, cost and quality. Both measures should be given equal importance as they are dependent on each other.
3.8 Accuracy of estimates

An estimate is some sort of approximation based on available information at the time and/or experience with the recognition of that relevant facts might be excluded. An example could be test data, where the only information available is that something has been successfully tested and that what where being tested passed the test. If one does not know how it has been tested and if it has been performed in the same way for each test, an estimate of the tests reliability needs to be made to determine how to assess its result, e.g. in 8 out of 10 cases the testing has been conducted properly.

The problems with generating reliable and consistent estimates are the inconsistency of the process (Galorath & Evans, 2006). The inconsistency comes from different opinions of what the basic elements of the estimates actually are and how they should be represented. Additionally, the individual related to the estimation process influences it through motivation, experience, training and biases. In addition to this, it is often hard to receive acceptance from management and customers when presenting realistic estimates. This is due to that there are always constraints and certain goals or limitations that must be met, e.g. there is a certain cost and time required to provide the needed functionality and be able to produce quality output, which might not be in an alignment with the realistic estimate (Galorath & Evans, 2006).

A good way to cope with this is by approaching the estimation process by collecting and evaluating historical information which provides the estimators a quantitative way of determining what has been happening in the past. Thus one can project what will happen in the future with at some degree of accuracy. To successfully realise this, it is crucial that organisations establishes fundamental project metrics that can be collected for every project. This should then in turn be evaluated and compared with each other, which then can be applied in the estimation process of future scenarios (Galorath & Evans, 2006).

The usability and validity of an estimate is determined by its degree of detail and the estimates range of uncertainty. To handle the range of uncertainty all estimate inputs should be classified as least, likely or most and not as a single fixed point. This is true even if the scope of the project is unknown, as using ranges for inputs can bound the problem which in turn simplifies the development of estimates that can be used for planning purposes. To ensure consistency in the estimation process, certain data is needed to be available from the start. This core information will allow the analyst to generate valid estimate outputs conducted in an efficient and effective manner. Not all of the needed information can be relied upon to be available from start nor that it will be collect from a single point. One most reach out and extract the information that people or processes associated with what one is trying to estimate possess (Galorath & Evans, 2006).

Another way of approaching the uncertainty of estimated indices are to work with confidence intervals and to perform tests. A confidence interval is according to Montgomery (2009) “An interval estimate of a parameter is the interval between two statistics that include the true value of the parameter with some probability” and is commonly used in statistics.

To be able to generate reliable results, the measurement accuracy of what is being intended to measure, is crucial. To ensure the accuracy, it is important that the equipment used is capable and properly calibrated for the particular task. One can divide the measurement error into five different error sources; Thoroughness, Repeatability, Reproducibility, Stability and Linearity. Here it is important that the inherent variation is small enough with regards to the tolerance limits, i.e. the repeatability should be high. The
reproducibility error source can be explained by the previously mentioned argument for repeatability. The difference between two measurements on the small thing at different time periods needs to be small enough, i.e. the stability needs to be sufficiently high. The linearity, i.e. the difference in security, of the entire measured interval needs to be examined. If there is no routines of how to handle these concepts there the uncertainty will be too great in order to make reliable predictions based on what has been measured.
4. Description of current state

In this section the overall process of field testing with its prior and post activities are presented together with all the different sources of data. The reader is introduced to the section at Scania in charge of both the testing conducted in-house, the field testing and how they utilise the data available for them.

4.1 General empirical information

This section present the general empirical information that has been collected in order to get a holistic understanding of the environment of the field testing.

4.1.1 YSPV – Quality Integration and Test

YS is a collective name for several different departments at Scania, Södertälje, that are associated with the after-sales service of vehicles. YSPV is a group under YS started 2/5-2013 with responsibilities in quality, integration and test. Their task is to handle the quality assurance and quality control of all the applications created by YS for the after sales market regarding troubleshooting, reparation, updates of software, i.e. electrical control units (ECU), and rebuilds of vehicles. The future goal is to include all YS products and services for workshops and service centres. YSPVs stakeholders can be seen in Figure 10 below, which are; service centres that uses the software and conducts conversions, field test workshops, method engineers, system owners and translators of software in R&D and the component developers. Service centres and field testers are the external stakeholders to both YSPV and the other internal stakeholders to YSPV. YSPVs relation to the internal stakeholders is mainly through testing and verifying that what they have developed works as intended. However, all the internal parties are in constant contact with each other and in this way ideas and information is exchanged, thus YSPV contributes with input to all internal parties. YSPVs main role is though as the last quality control unit of the developed software before it is released to the external stakeholders.
4.1.2 Service and Diagnostic Program 3 (SDP3)

Scania is using the software Service and Diagnostic Program 3 (SDP3) to troubleshoot and program the CAN-based part of the electrical system in their vehicles. SDP3 is also used by mechanics that troubleshoots Scania vehicles at service centres around the globe. The information in SDP3 is divided into two different categories of how one can search and interact with the electrical system, User Functions View and Electrical System View. In the Electrical System View one can see the different components and how they are connected, where an example of this could be that one can see that the speed sensor is connected to the recording equipment. To conduct a troubleshooting in the Electrical System View one can check circuit diagrams, read in- and out signals and activating components, i.e. looking at component level. In the User Functions View one can treat the electrical system by looking at different user functions, where a user function can utilise more than one control unit and several different components. In order to troubleshoot the electrical system through the User Functions view one looks at function diagrams and practically test the user functions. An example could be to check that the recording equipment sends information regarding the velocity to the combination instrument that displays it on the speedometer. A new category called Functions has been added from SDP3 version 2.18, which is a more user friendly method of the User Functions view. This category is still under development and has not replaced the User Functions view as yet.

SDP3 can be utilised for testing Scania’s trucks in the P-, G-, R-, T-series, busses in the F-, K-, N-series and industrial and marine engines with an engine management system (EMS). To perform a test one must
connect a computer with the testing program SDP3 to the vehicle, which is done with the vehicle communication interfaces (VCI2 or VCI3). In order to successfully connect, a USB-key needs to be attached to the computer, which grants full access to SDP3 and allows unlimited testing. When the software is connected, SDP3 reads the specification of the vehicle and based on this reading presents the chassis specific information together with all the stored failure codes. The connection process can be seen in Figure 11 and a visualisation can be seen in Figure 12.

![Diagram](image)

**Figure 11: SDP3 connection order**

There are two different ways to connect to a vehicle in order to perform tests, either through a wired connection or through a wireless connection, where VCI2 and VCI3 are the available vehicle connection tools. VCI2 and VCI3 can both be used to connect with a wire, while VCI3 also can connect wirelessly. When using the wired connection, there needs to be one computer for each vehicle and connection, whilst for the wireless connection one computer can connect to different vehicles from the same spot, however only one vehicle at a time. This way the mechanics or testers can conduct tests on different vehicles faster. When connecting to a vehicle, SDP3 reads the vehicle specifications and then only displays information that is relevant to the connected vehicle. The vehicle information is stored in a data file known as the Scania On board Product Specification file or SOPS file for short.
4.1.2.1 SDP3 job types

There are five different job types in SDP3 that can be used when troubleshooting with the diagnostic tool; Checks and Adjustments, Conversion, Maintenance, Product updates and Bodywork. These job types enable different sorts of troubleshooting and/or programming of the vehicle to be done.

Checks and Adjustments: This is the main work option which includes most of the functions available in SDP3 and is thus the most commonly used job type. It can be used to access information and read fault codes to troubleshoot vehicles through the different control units as well as from the functions point of view. It can also be used to change the adjustable values in the control units and conduct calibrations and resetting. Checks and adjustments can also help analyse the operational data of the vehicle. All the control units associated with a vehicle can be diagnosed in this job type and functions included within maintenance and bodywork can be conducted here.

Conversion: This work option is used to perform per-ordered updates and changes to the connected vehicle. It is used to update the SOPS file when any piece of hardware is changed in the vehicle, for example if the radio is changed, a conversion is performed to update the SOPS file with information about the new radio so that the vehicle is aware of the change that has been made. Conversions are usually bigger updates or changes made that require the SOPS file to be updated.

Maintenance: This work option is used when performing regular scheduled maintenance on a vehicle. It can only be used for checking various control unit parameters and reading fault codes, but it cannot be used to make any adjustments or changes to the control units. In other words, it is used to perform an inspection of the vehicle.

Product updates: This work option is used to update the software of the connected vehicle. These updates are smaller than conversions and may not require the update of the SOPS file. Product updates can be performed on control units, SOPS files and adaptation at engine renewal. Certain product updates can be mandatory to perform while others are optional.

Bodywork: This work option is used to access the additional bodywork added to the chassis of a vehicle. It includes complete checking functionality, but the ability to adjust parameters is limited based on the bodywork involved. Newer vehicles have a control unit named Bodywork Communication Interface which enable the functionality of this job type to be performed through the job type Checks and adjustments.

4.2 Process mapping

The processes involved in and around field testing are mapped and described in this section to be a deeper understanding of how the work is conducted.
4.2.1 Testing process

The overall process of testing consists of two main activities after the software is ready, In-house testing and Field testing, as can be seen in Figure 13. This information regarding the testing process has been collected through formal and informal interviews and discussions with Scania employees. The initialisation of the testing is triggered by the time plan that the release coordinator is responsible for. The time plan is put together in collaboration with the release coordinator and the different teams involved in the development of SDP3 which then gets distributed to the test leader at YSPV. It is the test leader’s responsibility to communicate the time plan to the test group and make sure that the tests are conducted in accordance with the time plan. The time plan consists of different components that together creates the product, i.e. SDP3. These components receives a prioritisation order of when they are to be finished and carried out. Then as a certain degree of maturity of the product has been reached the test plan can start to be developed.

4.2.1.1 In-house testing

The test plan is based on the release report of the software version from component developer which contain the changes made in the framework. Based on this release report, a mind map of the changes is created and the test plan is prepared accordingly in the test management system. The test leader then assigns the different tests that are to be conducted to the test members at YSPV who then carry them out. Below in Figure 14 one can see the major steps included from creating the time plan to the test plan.

The tests are conducted in three different ways, where the testers can use vehicles, rigs or demo files. Scania has a fleet of trucks and buses that are available for testing and these vehicles are used by the test team at YSPV. Testing on a vehicle is the most time consuming option because one must physically approach and connect the equipment to the vehicle, but it is also the most powerful as all available tests
can be conducted on a vehicle. The only limitation of this option, which is the limitation of the entire testing process, is that only the available vehicle configurations can be tested. Thus particular components that might be unique to vehicles with other configurations cannot be tested. To perform the desired tests on these vehicles, one needs to connect to the vehicle in question, which is attempted through the field tests that are described later. The same goes for the rigs, which require a physical connection but they are located in the office next to the testers, which simplifies and makes the testing more convenient. However, the rigs only consist of the most commonly used components which is a small part of all the available components in an actual vehicle. The last option is the demo files which are preloaded vehicle cases that simulate a connection to a vehicle and can be used to conduct tests. These tests are limited to the program, i.e. in this testing mode, no actual changes of the vehicle can be conducted. The demo files are accessed within SDP3 and are thus the quickest testing option.

Certain safety and critical tests are conducted for each new release during the testing plan. These tests ensure that there is no serious error in the software that need to be addressed immediately. Once these tests are passed, various positive and negative tests are conducted. Positive tests ensure that a function works as intended when all the operating conditions are met, while the negative tests make sure that functions do not run if the operating conditions are not within the specified limits. The test plan consists of two different phases, the sprint phase and the release phase. The sprint phase consists in turn of three to four sprints, where each sprint contains a newer release of the software which comprises of changes and new additions from the previous version of SDP3 that needs to be tested. Additionally, the sprint phase is divided into two separate parts. The first part of the sprint phase is the development and functional growth of the software (DFG), i.e. the new additions to the software and the changes made according to change requests (CR) that have been delivered in the sprint. Functional and regression tests (FRT) are conducted to ensure that all functions that were working in earlier versions are still working and to verify that the new additions are working as intended. The second part consists of the deviations that have been resolved, i.e. the trouble reports (TR) that have been corrected since the previous version. These TRs are tested and verified to ensure that they are fixed. If any errors or unexpected behaviour is encountered, the testers determine the root cause of the problem and create a TR in the JIRA, the problem database. The TRs created in JIRA are prioritised at meetings held twice a week and the selected TRs are sent to the responsible group to be fixed for the next release, which are then tested in the sprint for that respective release.

When both parts of the sprint testing has been done there is a delivery meeting where all the providers of components and changes of the products meet to present what they have done and what has been tested for the release of this product. This is where the second phase of the test plan is initiated, the release phase. At this point in time the “functional growth” of the software is completed, i.e. development of product is done, which in theory means that nothing new should be added or changed from that date until the release of the product. From this point the only focus is to make sure that the already applied changes work as attended for the release date of the product and that there are no major issues. This is however not how it works in reality as the language packages are added on during this phase. See Figure 15 for an overview of the test plan process
4.2.1.2 Field testing
Before the software can be sent out to the field testers in workshops for field testing, certain prerequisites must be fulfilled. First, the software must pass certain safety and critical tests. These safety tests ensure a certain level of quality to prevent any major issues during field tests because these tests are performed during actual service of real vehicles that are running on the road and any error could cause an accident.
Next, a risk analysis should be conducted to ensure that the major issues have been solved and that it is safe for field testing. Finally, the basic functionality of the software should be running smoothly without any known issues. These include reading active and inactive fault codes, adjusting parameters for the most important ECUs, spare part programming and conducting ECU updates through both, internal and external networks.

Once the prerequisites are met, the software is released for field tests in three stages as the different language packages are added to the software, the first stage is only with Swedish, the second stage is with Swedish and English, and the last phase is with all 18 languages. However, the releases do not always follow these stages in chronological order, where an example could be that the first stage is skipped and the second stage is initiated from the start. The number of testers in each stage varies, depending on the language that they use. A minimum of one week (five working days) is observed between releases in each stage when quick fixes are needed, otherwise the planned time range between releases in each stage is two weeks (ten working days). This time range can also be exceeded if deemed needed for a stage.

Various information is sent out to the field testers for each test. This information includes:

- previous issues that have been fixed in this software version. These issues should be tested.
- changes made in this software version. These areas should be focused on while testing.
- known issues in this software version. These issues should not be reported again.
- instructions on how to send in bug reports.

There could also be information to look out for certain vehicles with particular configurations that the software needs to be tested on. If these vehicles come in for service, they should be reported to the field test team and the details of the testing on these vehicles should be sent in.

If any unexpected behaviour is encountered, the field testers follow specified instructions to submit an error report. This includes submitting a screenshot of the problem, a description of the problem and under what condition it occurs, and the log files for the session to the field test team. The field test team first checks with the concerned group (component developers or method engineer) or in JIRA if the error is a known problem. If not, a TR is created in JIRA with the label field test for easy identification. These TRs are handled the same way as other TRs and are considered during the meeting held twice a week. The field tester is informed of the progress and possible solutions as soon as possible.

Usage data showing how many times each version of the software has been used on a daily basis by field testers is collected at the end of every week by the field test team. This data is compared with the number of uses per week from previous releases to get an idea of how well the field test has gone and determine if the field testing has been sufficient. After the third stage of field testing which includes all languages, the fourth stage comes, known as the stability period. During this period, no big issues are expected as this is the version that is to be released to the market. A problem found during this period would lead to delaying the release, if it is deemed serious. After the field tests are conducted to a satisfactory level, the software is ready to be released to the market. The overall test process can be seen in Figure 16.
Figure 16: Overall test process for SDP3 development
4.3 Available data

*Here all the available data sources with its corresponding data is presented. There is a short general explanation of each data source followed by the explicit data that can be derived from the source. This answers the first research question “What data can be collected during a field test of a diagnostic program?”*

4.3.1 JIRA – an issue tracking tool

JIRA is an issue tracking tool used by Scania to store and track TRs and CRs created during software development, maintenance and projects. TRs and CRs can be tracked from creation onwards, through all statuses such as open, in progress, resolved, closed, reopened and postponed. The description of an issue gives a clear picture of the problem and also includes information such as priority, affected software versions and the ability to attach documents to provide additional information. Comments can be added to an issue to give details on the progress of its resolution as well as communicate between the different personnel involved with the issue. Similar or connected issues can be linked to each other so that their information can be shared between them and also to enable them to be solved simultaneously. An issue can also be seen after it has been resolved to gain feedback and backtrack to see how the issue was resolved.

All TRs found and CRs created during the development of SDP3 before its official release to users is stored and tracked in JIRA. These issues are assigned to responsible personnel and solved on a priority basis. All TRs and CRs from field testing are found here under the label ‘field_test’.

4.3.1.1 Description of available data

In Appendix B one can see how the information in JIRA is displayed where the details of an issue includes; Type, Priority, Affects versions, Components, Labels, Affected projects, Epic link, Sprint, Status, Resolution and Fix version. Then there is a textual description of the encountered issue where the one filing the issue explains it based on the particular case. Thus different information can and is being included here. In addition to this, the issuer attaches log files and screenshots of the issue. Lastly, each issue contains a comment section where all involved parties at Scania communicates the progress of the issue and cooperates to solve it.

4.3.2 FRAS – a product quality follow up system

Scania uses a global system to record and follow up on product quality and quality deviations, called follow-up report administration system (FRAS) (Scania, 2014). All issues regarding a Scania product experienced by a customer are recorded and followed up here. FRAS includes applications such as failure reports, assignment, technical support and pre-sales helpdesk, long term testing, field test, deviation report, event, etc. All reported issues encountered by customers while running a released version of SDP3 are followed up and solved through FRAS. Each issue is assigned to a responsible group and leader who determine the appropriate actions to be taken. Each action related to the issue is documented in FRAS to give a high level of traceability.

4.3.2.1 Description of available data

In Appendix C one can see an example containing typical information from FRAS. It includes a brief description of each issue in the search field, where the unique id of the issue and the one responsible for it can be seen. When selection an issue one get even more detailed information where the Type, Design area, Phase, Address, Assigned group, Process and relevant meeting dates, affected product, time and
action plan, solutions, notes of the issue is available. Then there also is a more elaborative description of the issue in the description box, which contains diverse information depending on the issue.

4.3.3 Testlink - a test management system
Testlink is an open source test management system used by the testers at YSPV to manage their test cases and test plan. Information about all past and current test plans can be found here. Test cases pertaining to a similar function or category are grouped together under a test suite. All the possible tests, with instructions on how to perform them including the preconditions required, are included within Testlink and the test leader determines which tests are to be run in each test plan. The chosen tests are then assigned to various testers who carry them out. The testers use Testlink to note and track their progress during the execution of a test plan. Testlink gives information about what tests have been completed, who has performed each test, what was tested in each build of the software, priority of each test, how many of the scheduled tests were completed and the status of the tests. It also offers a comparison of different test cases to understand how each test case has gone and identify subtle differences.

4.3.3.1 Description of available data
Appendix D shows an extract of the data that can be exported from Testlink. The selected test suites and the different test cases in each test suite can be found here. The platform that each test case is to be performed on is mentioned along with its priority. The status, i.e. passed, not run, failed or blocked, of the test cases in each build of the selected SDP3 version is shown as well as the status from the last build of the version and the status from the last execution of the test case within the selected version. The status passed means that the test has been carried out as planned and that it passed it, not run means that the test has not been carried out, failed means that the test has been carried out but not successfully and blocked means that the test cannot be carried out due to that either a previous test has not been successfully run or that a test has not been run at all. While Appendix D shows only three builds of the SDP3 2.20 as an example, the data is available for all the builds of a version.

4.3.4 Cognos – a business intelligence tool
IBM Cognos Business Intelligence is a business intelligence tool by IBM that is used for reporting, monitoring, data analysis and data visualisation. It is used by Scania for a number of purposes including the tracking of conversions and product updates performed during the usage of SDP3. It contains data about all conversions and product updates performed through SDP3, no matter if it is conducted internally or externally.

4.3.4.1 Description of available data
An extract of the conversion and product update data can be seen in Appendix E. The order type shows whether it is a conversion or product update, with rebuild referring to a conversion while software corresponds to a product update. The client version is the SDP3 version used for the update while the ECU Family and ECU Scomm Id display the ECU and component on which the update was carried out respectively. The PSM version of the SDP3 version, dealer country code, vehicle identification number (VIN), serial number (the last 7 digits of the VIN), status and the date of creation of the update can also be found here.

4.3.5 Qlikview – a business intelligence tool
Qlikview is a business intelligence tool developed by Qlik and used for data visualisation. It is used by various departments at Scania to process and visualise key business data in an informative manner in
order to enable quick and informed decision-making. Currently, every click performed during an SDP3 session for certain users (internal users and field testers, which is recorded in the usage logs of a session, is analysed through Qlikview to display statistical information about the usage. This also includes information regarding the running environment such as the operating system, system language, SDP3 version, SDP3 language etc. as can be seen in Appendix F. Useful information about the common operations performed in SDP3 can be inferred from Qlikview.

4.3.6 Jenkins – a continuous integration tool
Jenkins is an open source application used for continuous integration during software development. It is used for continuous tracking and integrating of changes and testing of software builds during its development life cycle. It is used during the development of SDP3. All the different changes performed during the development of the software can be recovered from Jenkins, including the all the changes implemented between different builds.

4.3.7 SDP3 usage web application – a visualisation tool
SDP3 usage web application is a tool used by the field test team to display information about the usage of SDP3 during a field test. Currently, it is the only information being used to assess the result of field testing. It contains information about the number of uses along with the number of unique clients for each build of SDP3 and is available for time units as small as one day. It also includes the capability to compare the number of uses of different builds of SDP3 graphically. Examples of the data from the SDP3 usage web application can be seen in Appendix G.
5. Situation analysis

The overall data that has been collected from the data source presented in the previous chapter, beside the product updates data is summarised and can be seen in Appendix H, from which the following situation analysis data is derived from. Thus this chapter treats the research question “What attributes are relevant and the most important to consider when assessing the quality of a field test for a new release of a diagnostic program?”.

The amount of in-house testing for SDP3 version 2.19 to 2.22 can be seen in Figure 17 where significant differences between the versions can be observed. One reason for this is due to that YSPV started their work two years ago and consequently significant operational changes has been made which has an influence on the number of performed tests for the different SDP3 version over time. Thus the number of tests conducted per version is destined to vary, even though the frequency of operational changes can be expected to diminish over time. Furthermore, as a direct consequence of operational changes within YSPV, all desired data is not available for all the previous version of SDP3. This can be seen in Figure 17 as there is no number of conducted tests for 2.18. This does however not imply that there have been no in-house testing, only that it’s not recorded in the same way as the after that point.

For 2.20 and onwards, Figure 17 displays a decrease in the performed in-house tests whilst the amount of tests for 2.19 is significantly less. The low amount of testing for 2.19 is quite unexpected, which reason cannot be further explained than to be the outcome of the operational changes followed by introducing Testlink as a daily routine at YSPV in the time period of 2.19-2.20.

![In-house testing per version](image)

**Figure 17: In-house testing per version**

The amount of in-house testing is partly based on the number of changes applied to a SDP3 version. However, the amount of in-house testing does not necessarily represent the significance of the changes as a small change might need several tests to conform the change, whilst some larger changes of the software might only need one or a few test for confirmation. Also, there is a set of certain predetermined safety and critical tests performed for every version, which consequently affect the total number of tests
performed. Thus, the amount of changes in a version is determined by the number of TRs corrected in the version along with the number of CRs implemented.

The amount of fixed TRs and implemented CRs from 2.18 to 2.20 can be seen in Figure 18. Regarding the TRs one can see an increase in 2.19 and 2.20 compared to the other versions which have a similar number of fixed TRs. The CRs seems to decrease slightly in the two latest versions.

![Changes made per version](image)

**Figure 18: Fixed TRs and implemented CRs per version**

The number of uses during field testing is an important parameter to assess the field test. In particular, the number of uses of the last build or release candidate is especially significant because this is the version that is released. In some cases, if a big problem is found, the last build could have small changes to fix these issues and can be considered to be relatively similar to the previous build (second last build). In this case, we consider 75% of the uses of the previous build in addition to the last build. Thus, the last build for our consideration is the last build and, if only minor changes were made to the previous build, 75% of the previous build. The number of uses during field testing for each version can be seen in Figure 19 along with the proportion of the last build uses.
The field testing for each version of SDP3 results in a number of issues and problems. These issues and problems are made into TRs if they are unknown previously and not a solitary issue. The TRs are given severity ratings depending on their likely impact. The TRs found per version, classified by severity are shown in Figure 20 where it can be seen that the number of issues found during the field testing of 2.20 was very high compared to the other versions.

The long time period of field testing for 2.20 could explain the high number of issues found during field testing in for that version, which one can see in Figure 20. Furthermore one can see that the total number of weeks where field testing is conducted is fairly similar with the exception of 2.20 where 16 weeks of field testing passed.
In Table 2 one can see a noticeable correlation of 0.672 with the number of issues found after release with the number of blocker issues found during field testing, which indicates that the most important bugs that are found, i.e. Blockers, is related to the overall quality of a version. Otherwise the correlation found is not strong enough to argue for a reliable connection.

Table 2: Correlation table with issue severity and no. of issues found after release

<table>
<thead>
<tr>
<th>No. of issues found after release</th>
<th>Major</th>
<th>Critical</th>
<th>Blocker</th>
<th>None/Minor</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.366774</td>
<td>-0.155126</td>
<td>0.672214</td>
<td>-0.4702668</td>
</tr>
</tbody>
</table>

Despite thorough in-house and field testing, various issues are experienced by users after the release of the software. The aim of in-house and field testing is to minimise the number of these issues, which can be seen in Figure 22.
Correlation tests are conducted. Correlation provides the dependability between variables, which in turn can be exploited to get an indication through their predictive relationship. We suspect a correlation between the amount of change in a version, the field test uses, field test users, TRs found during field testing and the number of issues after release. Table 3 displays the strengths of correlation values, where a positive correlation means that if parameter X increase so will Y, while a negative correlation means that if parameter X increase, Y will decrease (Quinnipiac University, 2015).

**Table 3: Correlation values**

<table>
<thead>
<tr>
<th>Correlation</th>
<th>Weak</th>
<th>Moderate</th>
<th>Strong</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>&lt; 0.30</td>
<td>0.30 to 0.50</td>
<td>&lt; 0.50</td>
</tr>
<tr>
<td>Negative</td>
<td>&gt; - 0.30</td>
<td>- 0.30 to - 0.50</td>
<td>&gt; - 0.50</td>
</tr>
</tbody>
</table>

The following scientific hypothesis are formulated regarding the correlation between these parameters.

H1: The number of TRs found during a field test is dependent on the number of TRs fixed in that version.

H2: The number of issues found after release is dependent on the number of TRs fixed in that version.

H3: The number of TRs found during a field test is dependent on the number of CRs implemented in that version.

H4: The number of issues found after release is dependent on the number of CRs implemented in that version.

H5: The number of TRs found during a field test is dependent on the number of uses during the field test.

H6: The number of issues found after release is dependent on the number of uses during the field test.

H7: The number of TRs found during a field test is dependent on the number of users during the field test.
H8: The number of issues found after release is dependent on the number of users during the field test.
H9: The number uses during the field test is dependent on the number of field testing weeks.
H10: The number of TRs found during a field test is dependent on the number of field testing weeks.
H11: The number of issues found after release is dependent on the number of changes made during the release phase.
H12: The number of issues found after release is dependent on the number of uses of the last build of the field test.
H13: The number of issues found after release is dependent on the total number of implemented changes.

The presented hypothesis is tested based on their belonging parameters’ correlation. A summarise table of this data can be seen in Appendix I. In Table 4 one can see the outcome of each hypothesis, whether or not it is true or false, and its belonging degree of correlation. The correlation has been validate as plausible or at least possible by the Field Test Coordinator and Test Leader at Scania.

**Table 4: Hypothesis result**

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Result</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>H1: The number of TRs found during a field test is dependent on the number of TRs fixed in that version.</td>
<td>True</td>
<td>0.940</td>
</tr>
<tr>
<td>H2: The number of issues found after release is dependent on the number of TRs fixed in that version.</td>
<td>False</td>
<td>0.236</td>
</tr>
<tr>
<td>H3: The number of TRs found during a field test is dependent on the number of CRs implemented in that version.</td>
<td>False</td>
<td>0.084</td>
</tr>
<tr>
<td>H4: The number of issues found after release is dependent on the number of CRs implemented in that version.</td>
<td>True</td>
<td>0.906</td>
</tr>
<tr>
<td>H5: The number of TRs found during a field test is dependent on the number of uses during the field test.</td>
<td>Partly</td>
<td>0.611</td>
</tr>
<tr>
<td>H6: The number of issues found after release is dependent on the number of uses during the field test.</td>
<td>False</td>
<td>-0.307</td>
</tr>
<tr>
<td>H7: The number of TRs found during a field test is dependent on the number of users during the field test.</td>
<td>True</td>
<td>0.865</td>
</tr>
<tr>
<td>H8: The number of issues found after release is dependent on the number of users during the field test.</td>
<td>False</td>
<td>0.223</td>
</tr>
<tr>
<td>H9: The number uses during the field test is dependent on the number of field testing weeks</td>
<td>True</td>
<td>0.693</td>
</tr>
<tr>
<td>H10: The number of TRs found during a field test is dependent on the number of field testing weeks.</td>
<td>True</td>
<td>0.772</td>
</tr>
<tr>
<td>H11: The number of issues found after release is dependent on the number of changes made during the release phase.</td>
<td>False (insufficient data)</td>
<td>-0.934</td>
</tr>
<tr>
<td>H12: The number of issues found after release is dependent on the number of uses of the last build of the field test.</td>
<td>True</td>
<td>-0.708</td>
</tr>
<tr>
<td>H13: The number of issues found after release is dependent on the total number of implemented changes.</td>
<td>False</td>
<td>0.495</td>
</tr>
</tbody>
</table>
5.1 Product updates

In Figure 23 one can see the development of the product updates during the field testing, where the number of product updates carried out is displayed for each version, together with its corresponding success and failure rates. The reason for 2.20.1 being so significantly less than the others is due to that it is only a minor extra build release for 2.20. This build was released due to, as previously mentioned, the many complications experienced for that version. Even if 2.20.1 did not get field tested properly, which can be seen on the number of product updates in Figure 23 and is, it is included here due to its actual usage after release, which is comparable with 2.20.0, but it is not used during the analysis of the field testing data. Furthermore, the reason for not including data regarding 2.18 and 2.19 is due to that the database Cognos where all product updates data is retrieved from, did not collect this data until 2.20 was release, and thus only these version are displayed and analysed. In addition to this, one can see that the trend regarding the done and fail ratio seems to be improving with each version release.

![Field testing product updates](image)

**Figure 23: Development of the product updates for field testing**

In Figure 24 one can see the same data as displayed in Figure 23, but for after release. The number of updates for 2.21 is relatively high because 2.20 had two releases, 2.20.0 and 2.20.1, thus splitting their uses, while the number of updates for 2.22 is low due to the fact that it was released less than one month prior to the data collection. The trend of the done and fail ratio does not display any clear direction, even
though 2.22 displays a small improvement one cannot draw reliable conclusions based on the current data since it is only a portion of the whole, future data set.

Figure 24: Development of the product updates after release

The distribution of the ECU updates performed per version during field testing can be seen in Figure 25. The different versions exhibit a similar distribution of updates, with a few ECUs, such as X and Y, exhibiting greater variations due to changes in the available product updates during the use of different versions.

Figure 25: Proportion of updates during field testing per ECU per version

Figure 26 also displays the distribution of the ECU updates performed per version, but after release. The distribution of the ECUs is closer between the versions here, than during field testing, with X and Z continuing to exhibit a higher variation. In total the pattern of the usage between field testing and after release displays significant similarities.
Figure 26, Figure 27 and Figure 28 show a comparison of the proportion of updates performed on each ECU during field testing and after release for the different versions. It can be seen that the field test is a good representation of the actual usage of the product updates after release.
Figure 28: Proportion of updates per ECU in field test and after release in 2.21

Figure 29: Proportion of updates per ECU in field test and after release in 2.22

Figure 30, Figure 31 and Figure 32 show a comparison of the failure rate of the updates per ECU during field testing and after release for the different versions. It can be seen that the failure rate during the field testing of 2.20 was higher than after release, with it being twice as much for many ECUs. The failure rate during the field testing of 2.21 and 2.22 show a similar pattern with that of the failure rate after release. It should be kept in mind however that the sample size for some of the ECUs during field testing is low, and conclusions drawn from this data must be considered with caution.
Figure 30: Failure rate of updates per ECU in field test and after release in 2.20

Figure 31: Failure rate of updates per ECU in field test and after release in 2.21
In Table 5 one can see failure data for both field testing and after release, but due to confidentiality the different ECU families with its belonging ECUs needs to be censored. In after release one can see the range from which the failure ratio for the different version contains themselves within. The biggest range is displayed by one single ECU X which stands for the entire data set in that family. Besides this ECU X all other ECUs has a failure ratio with a range lesser than five percent. Furthermore, one can see the average failure ratio for each ECU in total for 2.20.0, 2.20.1, 2.21.0 and 2.22.0. The Average value of product updates that are less than 200 for each individual version should not be assessed. This since the overall usage per ECU is in number of thousands, and is thus excluded to not give an incorrect picture of reality due to its high weight it would corresponds to with such few data points. The value for ECU X previously mentioned stands out regarding the average failure rate for after release and is also the ECU that has the widest range value. A censored ECU’s distribution that represents the most failures can be seen in Appendix J, where there is one that stands out significantly.

The tab field testing contains the average failure ratio for each ECU family and specific ECU during field testing for 2.20.0, 2.20.1, 2.21.0 and 2.22.0. The ECUs with a usage of more than 50 product updates on average between the previously mentioned versions are deemed as useful data. This due to that a smaller number cannot give reliable values. The column Total Number displays the total number of product updates made during field testing for the previously mentioned versions.
Table 5: ECU comparison between field testing and after release

<table>
<thead>
<tr>
<th>Field Testing</th>
<th>Average failure %</th>
<th>Range of failure %</th>
<th>Total ECU</th>
<th>After Release</th>
<th>Average failure %</th>
<th>Range of failure %</th>
<th>Total Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>17%</td>
<td>0%</td>
<td>297</td>
<td></td>
<td>1,5%</td>
<td>0%</td>
<td>31435</td>
</tr>
<tr>
<td></td>
<td>0%</td>
<td>15%</td>
<td>75</td>
<td></td>
<td>1,5%</td>
<td>0%</td>
<td>8262</td>
</tr>
<tr>
<td></td>
<td>12%</td>
<td>0%</td>
<td>396</td>
<td></td>
<td>4,8%</td>
<td>0%</td>
<td>53173</td>
</tr>
<tr>
<td></td>
<td>0%</td>
<td>0%</td>
<td>0</td>
<td></td>
<td>0,0%</td>
<td>0%</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td>6%</td>
<td>0%</td>
<td>250</td>
<td></td>
<td>3,6%</td>
<td>0%</td>
<td>41707</td>
</tr>
<tr>
<td></td>
<td>0%</td>
<td>0%</td>
<td>0</td>
<td></td>
<td>0,0%</td>
<td>0%</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td>9%</td>
<td>0%</td>
<td>108</td>
<td></td>
<td>12,2%</td>
<td>0%</td>
<td>19845</td>
</tr>
<tr>
<td></td>
<td>0%</td>
<td>0%</td>
<td>13</td>
<td></td>
<td>4,6%</td>
<td>0%</td>
<td>2084</td>
</tr>
</tbody>
</table>

The available data and the analysis of it presented above, presents a good base on which the frameworks can be developed. The next chapter discuss the goal of field testing and how a restricted and unrestricted framework of how to assess it can be developed and the steps of how it can be used is presented.
6. Framework development

In this chapter the research question “How should a method for assessing field testing be developed?” is answered and where the necessary parameters to collect are presented together with a discussion of their implications. The restricted framework is developed and its structure is presented stepwise and the unrestricted frameworks different additional parameters that are to be considered and how they can be utilised are presented.

The goal of the field testing is to detect the severe issues missed, if any exists, after the in-house testing has been conducted and thus give sufficient knowledge about whether or not to release the SDP3 version in question on time or if it needs to be postponed. Thus it is important to put the field testing in a holistic perspective to fully understand it, where both the process before and after is considered. Then as the final goal of the entire process, of which the field testing is a part of, is to release a software with as few bugs as possible the measurement to focus on is the number of issues found after release where this number consequently is to be minimised.

Then to better assess the field testing and be able to draw reliable conclusions regarding the quality of the software one must identify the available data and the needed/usable data in order perform a proper analysis. This way a base can be created that the assessment of the field test can rely upon. Then to both incorporate the available data and the data identified as useful, two different frameworks are developed, where the restricted framework contains all usable data at this point and the unrestricted framework uses all the desired identified data. The restricted framework consists of the overall changes made to a version, issues and usage during field testing and lastly the after release issues. These three main categories consists in turn of several different parameters that should be considered when assessing a field test of a diagnostic tool. The relevant parameters can be seen in Table 6 together with how the collected data for each parameter can be present per version.

Table 6: Data to consider when assessing the field test

<table>
<thead>
<tr>
<th></th>
<th>Version</th>
<th>...</th>
<th>2.20</th>
<th>2.21</th>
<th>2.22</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Implemented changes</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of Trouble Reports fixed</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of Change Requests implemented</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of TRs + CRs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Field testing</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of uses during field testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of unique clients</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of weeks of field testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of uses of release candidate and extra release</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of unique clients of release candidate and extra release</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of Trouble Reports found during field testing</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of changes during release phase (TRs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>After release</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of issues found after release (FQs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6.1 Implemented changes

To identify the changes implemented in a version one should collect the number of TRs and CRs implemented for that version. Then it is relevant to look at them both separately and together, since their
correlation with the number of issues found after release differ. However, according to the correlation, the number of CRs implemented for a version has the greatest connection with the number of issues found after release with a value of 0.906 and thus is the primary parameter to consider. The TRs fixed in a version displays a low positive correlation of 0.236 with the number of issues found after release and the joint value displays a moderately strong correlation of 0.495, which most likely is due to the high correlation for CRs implemented. However, the TRs fixed for a version is still interesting to consider as it displays a weak positive correlation and also partly is dependent on the field testing. This due to that if the field testing finds a bug and create a TR out of it, it does not appear as a new issue after release. Consequently the quality or range of the field testing play a significant role here. When it comes to the number of CRs implemented for a version, it seems logical that this parameter should bring the biggest software change and uncertainty into the software and consequently be correlated with the number of issues found after release. This since an implemented CR in general is a completely new addition of code to the software compared to a TR, which in general are a correction of already existing code. In a way, one can argue for that a CR implementation is a more significant change addition of the software than a TR is, and with every change the possibility of something being out of order increases. Thus one, in general, probably would expect more issues in total to be missed during the in-house testing and then to appear during the field testing and after release.

6.2 Field testing
There are different parameters that are interesting to consider for the restricted framework that falls under the main category Field testing, which one can see in Table 6. When considering the number of TRs found during field testing, one can see a strong positive correlation of 0.934 with the TRs fixed for the corresponding version. This suggests that the more fixed TRs in a version, the more TRs are found during field test. This would seems logical as the more TRs that has been fixed in a version the more changes in the code needs to have been applied, which consequently should increase the chance of encountering bugs that results in TRs. However, the initial thought and intuition suggests that fixing a TR, which is as mentioned above to correct already existing code that does not work as intended, would not result in such a high correlation. Nevertheless it is still a change and with every change the chance of something malfunctioning increases. However, the TRs fixed for a release could include the TRs found during the field testing of that version since the TRs found during field testing can be reported back to Scania and fixed within the same version if deemed important to fix, consequently increasing the correlation between the two parameters, but distorting the relationship between them since they are not mutually exclusive. Thus it cannot be concluded that the number of TRs that are found during field testing will increase if the number of TRs fixed before field testing increases.

Since the correlation with the TRs fixed for a version and the number of TRs found during field testing is so strong, a moderate correlation between CRs implemented for a version and the number of TRs found during field testing were expected. This is not the case and the correlation is very weak with a value of 0.084. One reason for this could be that CRs are something completely new in contrast with TRs, and that field testers in a sense is using what they always have been using. This way, field testing might not encounter the new functionalities brought with the implemented CRs in the same extent as with TRs where known functionalities have been fixed. However, with each release for field testing, a report with all the changes a forwarded, which implies that the field testers are informed regarding new editions to
the software. This could also be due to the fact that the TRs fixed for a version could include the TRs found during field testing, thus increasing the correlation between them but not with the CRs implemented.

The number of field testing weeks, i.e. time, shows a positive correlation of 0.693 with the number of field test usage, which is to be expected, while the total number of field test usage and number of TRs found during field testing have a moderately strong correlation of 0.611, which implies that the field testing is possibly finding problems that are missed during in-house testing due to the wider range of vehicles that are used at the workshops. This also implies that the TRs found during field testing is dependent on time, but could also be due to the number of builds released during field testing, as each new build contains changes, they are potentially bringing more bugs into the software and thus, with the change, the chance of a TR being filed is increased.

The initial thought regarding the number of total uses during field testing were that the more field testing usage the less issues found after release would appear. However, the correlation between the number of total uses during field testing and the number of issues found after release only display a weak negative correlation of -0.307. The reason for this is found to be that the total number of field test usage includes usage from all the builds of the version in question even though it is only the last build that finally is released to all customers. The weak correlation emerge since with each new build release, changes are implemented which makes the usage during the previous build less connected and thus less representative of the release candidate. This relationship becomes more apparent when looking at the correlation between the uses of the release candidate and the issues found after release. Here one can see a moderately strong negative correlation of -0.667, which implies that more uses of release candidate during field testing leads to less number of issues found after release, indicating that it is an important parameter to consider when assessing the field testing.

Since the number of last build uses and the number of issues found after release has a moderately strong negative correlation and the number of CRs implemented has a strong positive correlation with the number of issues found after release, a target value of the number of last build uses can be created based on the number of implemented CRs. This target value can be used to assess whether or not a version of SDP3 is of high quality enough to be released. i.e. since the above mentioned relationships, one can look at the number of CRs implemented and issues found after release and decide an acceptable number of issues found after release, then see what number of implemented CRs that that would correspond to. Then the number of issues found after release can be the base on which approximate number of last build uses should be aimed for in order to minimise the number of issues found after release.

6.3 Product updates
The product updates conducted during field testing gives an indication of what areas have been tested as well as the functionality of performing product updates through SDP3. As seen in the situation analysis, the product updates performed during field testing is a good representation of the usage after release. Thus, it can be considered while assessing a field test and used to get a picture of the possible use after release. Each successful update performed during field test strengthens the confidence that the update for that ECU is functioning as intended. It is essential to make sure that the updates for the commonly updated ECUs is working properly, thus the ECUs with more than 200 updates conducted that were identified earlier should be assessed.
To take the above mentioned parameters with their belonging level of impact in consideration to assess the field testing, the three following steps needs to be conducted in order for the framework work as intended. A sample of the actual design and appearance of the framework can be seen in Appendix K.

1. **Collect data**
   The first step is to collect the available data, which is an ongoing endeavour during the entire development process of the software. The data of interest to collect together with where it can be retrieved from can be seen in Table 7. One shall export the number of CRs and TRs from the database JIRA, which is done by filtering in JIRA with Project as SDP3 and Issue Type as Trouble Reports and Change Requests. Then export all the data to Excel. Regarding the product updates data, one shall ask the Cognos owner for Order type, Client version, Client PSM version, Dealercountrycode, VIN, SerialNo, ECUFamily, ECUScommid, Status Type, Creation Date and Vehicle. Then field testing usage data needs to be collect from SDP3 usage where the time duration of the field testing also is to be collected. From the database Jenkins all the changes during release phase is to be collected and lastly the issues found after release shall be collected from the database FRAS.

Table 7: Data collection table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Data source</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Trouble Reports fixed</td>
<td>JIRA data</td>
</tr>
<tr>
<td>No. of Change Requests implemented</td>
<td>JIRA data</td>
</tr>
<tr>
<td>Total no. of Trouble Reports fixed and Change Requests implemented</td>
<td>JIRA data</td>
</tr>
<tr>
<td>Total no. of uses during field testing</td>
<td>SDP3 Usage data</td>
</tr>
<tr>
<td>Total no. of unique clients</td>
<td>SDP3 Usage data</td>
</tr>
<tr>
<td>Total no. of weeks of field testing</td>
<td>SDP3 Usage data</td>
</tr>
<tr>
<td>Total no. of uses of release candidate and extra release</td>
<td>SDP3 Usage data</td>
</tr>
<tr>
<td>Total no. of unique clients of release candidate and extra release</td>
<td>SDP3 Usage data</td>
</tr>
<tr>
<td>No. of Trouble Reports found during field testing</td>
<td>JIRA data</td>
</tr>
<tr>
<td>No. of issues found after release (FQs)</td>
<td>FRAS data</td>
</tr>
<tr>
<td>No. of uses before release candidate</td>
<td>SDP3 Usage data</td>
</tr>
<tr>
<td>No. of changes during release phase (TRs)</td>
<td>Jenkins data</td>
</tr>
</tbody>
</table>
2. **Insert the data in the framework**

   When inserting the JIRA data one must delete all the previous JIRA data in the framework in the worksheet JiraData and then insert the new data there. The product update data from Cognos is to be added below the already existing data in the worksheet ECU_Update in the framework. The SDP3 usage data and the Jenkins data needs to be manually inserted in the worksheet InputData. Then as new data continuously is going to be available, one needs to updates the raw data at reoccurring intervals. When the framework is loaded with the desired data, it generates various graphs displaying the TRs and CRs implemented per version, total changes implemented per version, uses during field testing with both before and during the release candidate uses, the number of field testing weeks per version, issues after release together with the number of changes done during the release phase. Then regarding the product updates, graphs displaying the number of successful ECU updates per version, failed ECU updates per version, successful updates per vehicle type and the number of updates per ECU in total is displayed. One can see the previously mentioned graphs in Appendix L.

3. **Interpret the graphs**

   The last step is consequently to interpret the graphs, found in the worksheet Graphs in the framework, and with the help of recorded historical information and the derived correlation between the parameters of interest assess the development of the software during field testing. The graphs can be manipulated to display the different versions of interest to make the desired comparison in question. This is done in the worksheet MainSheet in the framework where one select the start and end version, which then displays graphs with data from all version in between and including the start and end version.

   These steps constitutes the foundation of the framework which creates a base on which the field testing can be assessed upon with the help of the parameters presented in Table 7. This framework provides a good overview of the different parameters that are affecting the field testing, where the most prominent result is the correlation found between the number of CRs implemented and the number of issues found after release, which in turn with its connection to last build uses enables a target value of the number last build uses to be based on the number of implemented CRs for that version. Thus this have been done in collaboration with the field test coordinator and the test leader at YSPV. Table 8 the number of implemented CRs and its corresponding number of last build uses that are to be aimed for can be seen.

<table>
<thead>
<tr>
<th>No. of CRs implemented</th>
<th>No. of last build uses required</th>
</tr>
</thead>
<tbody>
<tr>
<td>Below 35</td>
<td>350</td>
</tr>
<tr>
<td>35 to 40</td>
<td>375</td>
</tr>
<tr>
<td>40 to 45</td>
<td>400</td>
</tr>
<tr>
<td>45 to 50</td>
<td>425</td>
</tr>
<tr>
<td>50 to 55</td>
<td>450</td>
</tr>
<tr>
<td>above 55</td>
<td>500</td>
</tr>
</tbody>
</table>
6.5 Unrestricted framework

The unrestricted framework consists of an identical copy of the restricted framework plus additional aspects that we believe to be of interest when assessing a field test of a diagnostic tool that currently are not available to consider or utilise due to practical complications. Thus the following is suggestions of future work.

Today the program Qlikview is tracing and displaying every click made in SDP3, which is connected to a unique client and vehicle. This is the most in detailed information regarding the actual use of SDP3 that is available. However, Qlikview is collect its data from log files, where in theory much more information is possible to extract. A possible source where the actual usage and adjustments being made with the diagnostic program can be extracted from is the SDP3 Tool logs. However, in order to in a practical way, that enables this information to be utilised in the assessment of the field testing, a program or script needs to be created that can find and extract the use and adjustments being made. This way a more detailed base can be created as the actual specific usage can be analysed from which one can, in a better way, assess the field test. This way one can check the actual usage of all the implemented changes during the field test, which consequently would enable one to take better decision regarding the quality of the software. Furthermore, one can see the different languages that are being used and thus get an indication of where certain usage and adjustment in SDP3 are made. This would consequently enable one to pin point certain specific functions that one want to be tested during the field testing to those that in a great way utilities those function in their daily work, which then would be less of an effort for the field testers. It could also be the initialisation of detecting where certain failure occur, which then can be analysed and the root cause of the failure can better be identified and thus solved. Here it would be interesting to look further into the difference between vehicles, e.g. if there is any particular occurrences found only for trucks or buses. SDP3 tool logs could also reveal the vehicle connection type in a good way, which could shed light on whether or not a failure is due to the connection itself or if it is an actual bug in the software. At this point is just shown as a failure, no matter the actual cause. Thus the information embedded in the SDP3 tool logs is something we recommend Scania to look into in the future.

Regarding the product updates, one could construct control charts that considers the both the total use and the success and failure ratios in total and on ECU level in order to detect occurrences that are out of the ordinary. Thus Scania can get a first indication of if something is wrong, increased overall usage, changed proportional usage and thus the priority of different ECUs might need to be changed. However, this needs more data to be available in order to able to draw reliable conclusion around it. Thus it is a future recommendation for Scania to look into and consider when assessing the field testing.

Additionally, each TR are supposed to be assigned a severity level. Today that is not always the case, and when they receive a severity level, it is based on the one handling that particular TR and thus making it subjective. We believe, that if a standardised system for how to assign a severity level to each TR is constructed together with that the actual usage is extracted from the SDP3 tool logs, one could keep track of the field test usage of the implemented TRs with high severity and make sure that they work as intended for the release candidate.

For the unrestricted framework, there has been no actual framework developed. This as the appearance of the additions to the unrestricted framework from the restricted framework are unclear, the future framework needs to be developed as new data becomes available. This to ensure that it is fitted and utilised as good as possible with the already existing information.
7. Evaluation of the frameworks

In this chapter the two different frameworks are discussed and their use is pointed out. The different shortcomings of the findings are discussed, where its implications on the result and the future work that needs to be done in order to overcome them are presented.

The restricted framework is ready for use at present whilst the unrestricted framework needs further work in order to be utilised. This is the main difference between the two. Additionally, the restricted framework is only utilising data that is available at hand and that does not need any more powerful processing than can be handled in Microsoft Excel, from which the framework is developed. Thus the restricted framework uses the available data in order to give an indication and direction of the field testing being carried out from a more blunt view compared to the unrestricted framework which will provide information of a more detailed nature. Thus the recommendation is to look further into the matters described in the previous section.

The target value of the number of last build uses with regards to the number of implemented CRs are based on the expertise of the field test coordinator and the Test leader at YSPV, with regards to the tending trend of increasing field testing uses. However, these number might need to be adjusted in the future when more data is available. This is a shortcoming as data has not been collected in a standardised way for more than four to five versions. Depending on what data it is, it might not have been collect for more than two to three version, which consequently bring a lot of uncertainties into the assessment of the field testing. This as each new data set that becomes available can change the overall picture quite significantly, which advocates that the framework needs to be reviewed as it is provided with new data. The shortage of data is also present when calculating the correlation between presented parameters of interest, where H11 displays a negative correlation of -0,934, which checks if the number of issues found after release is dependent on the number of changes made during the release phase. However this data set only consists of three data points and thus makes it impossible to draw any conclusions regarding these parameters which consequently must be neglected.

The accuracy and usefulness of the framework will improve with time as more data becomes available. This is still a big improvement from how the field testing has been assessed previously though, where the assessment relied almost exclusively on gut feeling. The number of uses, unique clients and an overall view of the number of reported problems were taken into consideration, but without anything to base it on or that could give a target value of the needed number of uses.

The other parameters used in the framework is also providing a good overview where one can compare the development of the field testing and also the quality of the software. It gives an indication of the direction of it. An example is the number of in-house testing that can vary due to several distinct reasons, still the overall number is an indication of the amount of changes applied to a particular SDP3 version. Thus a useful measurement that can be used to compare the overall testing conducted between versions with regards to the amount of changes. Consequently, 2.20 can be assumed to have been implementing a larger amount of changes than the other versions displayed in Figure 17, which with all the facts available has its explanation.

After the in-house testing and field testing have been conducted one can track the amount of occurred failures of the software updates carried out after the SDP3 version in question has been released. This data reveals where, in terms of ECUs, the most failures have occurred since the release of the SDP3 version
up until present date. Thus it is possible to identify certain ECUs that have been functioning poorly, and thus could get extra attention during upcoming the in-house testing and the field testing.

The evaluation of the study can be summarised as follows

- The data taken into consideration is limited for many parameters and some results cannot be relied upon completely.
- The framework gives a good overall idea about the field test using the data available.
- The framework will be strengthened once more data from upcoming versions becomes available and is added to it.
- The unrestricted framework gives a good direction for future work that can be conducted to achieve a more detailed understanding of what has been tested during a field test.
8. Conclusion

This study analysed the field testing of a diagnostic tool and developed a framework to assess the field test as a quality control method for the diagnostic tool. The developed framework consists of two parts – one ready to be used immediately and the other requires additional future work. The framework that can be utilised right now considers data from before as well as after field testing to get a holistic view of the field test and also incorporate all the factors that influence the field or that the field test influences. The framework aggregates the various data that can be collected before, during and after a field test and calculates and displays the attributes that are relevant to consider to assess the quality of a new release of a diagnostic program. This study can be furthered by investigating the possibility of taking the clicks performed during usage, the tool logs and complete vehicle configuration information into account to get a more detailed picture of what exactly has been performed during the usage of the diagnostic tool.

The generated framework gives a good starting point to assess a field test and obtain a basic understanding of the quality of the diagnostic tool by understanding the field test in better detail. The result of this study will be strengthened further when more data becomes available and is added to the framework. Then the results points towards that the more changes implemented into a version of the software, the more issues or faults will appear. Then as the amount of usage during field testing is positively connected with the number of issues found, this relationship leads to that the more field testing conducted the more issues and faults will be discovered, and thus enables one to take action.

An important aspect is how one defines a good field test, if it is by finding many bugs or not. An example could be if field testing find many bugs that results in TRs, one might assume that the entire version contains several bugs which would show as issues found after release as well. However, this does not need to be the case. Here the quality of the field test becomes relevant to take into consideration as if the field testing is successful it will find the present bugs and thus new issues after release will be small. Furthermore, this does not imply that if the field testing results in few TRs that they have missed crucial bugs which will result in a lot of issues found after release. Consequently the assessment of the field testing can be troublesome. However, with the created framework, this assessment can be done with a better base as decision criteria.
9. References


10. Appendix

In this section all the appendices are presented that are referred to earlier in the paper.

Appendix A: Formal interviews

In interview A below one can see the interview questions asked to the test leader and one test team employee to deepen our understanding of the processes connected to the field testing.

Interview A

Questions to keep in mind

- What is good?
- What do you think can be improved?
- What is not so good?

1. What is your job at YSPV? –
2. How do you receive the information that it is time to start testing? – Start
3. What is the first thing that you do when a new software edition should be tested? – Start

4. Do you follow a standardised method/path when you engage a new edition? – Documentation
   - Is there a documented approach of how you shall engage a new testing? – Documentation
5. What kind of general steps do you conduct? – Overall process
   - What is included in each step, detailed description – Overall process

6. What data and information do you receive when a new edition is release that you shall conduct testing on? Information flow
   - How is the information that you receive transferred/communicated? Documents, orally, etc? – Info flow and documentation
   - Do you feel that you receive enough information for a new edition to start testing right away? – Information flow
   - If you need more information, where do you get that? Same place each time or different? What is the additional info that you need? Is there a plan for this? – Information flow and documentation
   - Do you ever feel that you cannot find the information you are looking for? Is it documented? – Documentation
   - Do you use information from failure reports that come from field testers from older editions?

7. Is there a standardised way or approach of how to conduct the testing? - Testing
8. What parameters are you testing? Software point of view, and outside the software, i.e. hardware? - Testing
   - Do you conduct different tests simultaneously? process

63
9. Do you make the same test on different vehicles? I.e. different trucks and also trucks to buses? Do you know to 100% that a change will affect all vehicles identically? – **Testing**

10. When you detect a fault during you testing, how is that treated? You make a handwritten note? In a word document? Is the notes available for only you? – **Documentation and information flow**
   - How do you determine if the tested component passes the test or not? Is there an interval or just pass or not pass? – **Testing**
   - When do you create a failure report? What are the qualifiers? – **Documentation / problem solving**

11. What happens after a fault report is created? What kind of options do you have to solve the problem? How do you solve it? – **Problem solving**

12. Is there somewhere you can find all you previous failure reports and notes? Do you use it? – **Documentation / information flow**

13. How do you document how a problem been solved? – **Documentation**

14. How do you follow up on the problem? – **Follow up**

15. Does it exist documentation of solved problems? - **Documentation**

16. Do you handle one failure at the time or do you handle several simultaneously? - **process**

17. Is the solution updated into the software immediately or does it wait for the next release? - **process**

18. How is the update conducted and by whom? BasApp? Can you do it yourself? - **process**

19. What is the next step after a problem has been solved? – **Problem solving / overall process**

20. Describe the entire process of how a solution is realised – **process / problem solving**

21. When is the testing of an edition complete? No failures left? What is the decision criteria? – **End**

22. Do you continue testing after the software goes out for field testing? – **end**

23. How did you conduct the testing of the last edition? Beginning to end - **Overall process in reality**

24. What happens during a field test? What is a field test according to you? – **process / problem solving**

---

**Interview B**

**Job types**

1. What is the Maintenance job type in SDP3?
2. What is the product updates job type in SDP3?
3. What is the Bodywork job type in SDP3?

**Testlink**

1. What is testlink?
2. Where can we get information of what and how much has been tested for a release, in-house? Can we track it? How? Do you have any performance metrics for the testing?

**Conversions and product updates**

1. Do you test conversion and product updates? Why do you/do not test it?
2. What info do you get from this? How can it be used/interpreted?

**PIDAT, vehicle info**

1. Where can we get info about the vehicles? What can we get from the VIN no.? What is PIDAT? Is that where you get vehicle info? How do we access it/get access?
Appendix B: Example of raw data from JIRA

SDP3 / SDP3-3286

FT - Stopped at the end of a succesful software update BCI

Details
Type: Trouble Report
Priority: 0 - None
Affects Version/s: SDP3 2.22 (SOP1503)
Component/s: BasApp
Labels: 11/2 9/2 FT_150303 field_test
Affected Project/s: SDP3
Epic Link: Trouble Reports
Sprint: Sprint 1 i 1510

Description
Rapport från fältet:

SDP3 2.22.0.292 stopped at the end of a succesful software update for the BWE, BCI1 control unit. The SDP3 window was blocked, and not active any button, see picture. SDP3 was neccessary to stop forced and restart.

Conversion Order
VIN:

See picture, demo and log files ziped.

Best Regards, Stefan Hermann

Attachments
Appendix C: FRAS raw data
# Appendix D: Testlink raw data

<table>
<thead>
<tr>
<th>Test Suite</th>
<th>Test Case</th>
<th>Platform</th>
<th>Priority</th>
<th>2.20 All Lan</th>
<th>2.20 SV</th>
<th>2.20 Sprint2</th>
<th>[Last Build]</th>
<th>Last execution</th>
</tr>
</thead>
<tbody>
<tr>
<td>01. Vera Adjust &amp; Guides</td>
<td>TC-1:Reducing the fuel pressure S8</td>
<td>Win 7 Internal</td>
<td>High</td>
<td>Passed [v1]</td>
<td>Passed [v1]</td>
<td>Not Run [v1]</td>
<td>Passed [v1]</td>
<td>Passed [v1]</td>
</tr>
</tbody>
</table>
### Appendix E: Cognos raw data

<table>
<thead>
<tr>
<th>Order</th>
<th>Type</th>
<th>Client Version</th>
<th>Client VM Version</th>
<th>Dealer/Country Code</th>
<th>VN</th>
<th>Switch No</th>
<th>IC/Family</th>
<th>IC/CommId</th>
<th>Status</th>
<th>Created Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_2_027</td>
<td>ESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 10, 2014 11:46:20 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_3_06</td>
<td>FR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 14, 2014 3:19:37 PM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>GS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 16, 2014 12:37:45 PM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_2_027</td>
<td>AT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Executing</td>
<td>mar 20, 2014 1:11:26 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_3_06</td>
<td>GS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 26, 2014 2:54:19 PM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 31, 2014 8:06:35 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_2_027</td>
<td>ESI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>apr 9, 2014 11:46:20 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_3_06</td>
<td>PT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Executing</td>
<td>apr 10, 2014 11:20:11 AM</td>
</tr>
<tr>
<td>2.1.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 405_2_17</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Failed</td>
<td>apr 28, 2014 2:14:46 PM</td>
</tr>
<tr>
<td>2.1.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 405_5_17</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Failed</td>
<td>apr 28, 2014 4:14:25 PM</td>
</tr>
<tr>
<td>2.1.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 405_2_17</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Failed</td>
<td>apr 28, 2014 4:14:25 PM</td>
</tr>
<tr>
<td>2.1.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 405_5_17</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Failed</td>
<td>apr 30, 2014 9:05:52 AM</td>
</tr>
<tr>
<td>2.1.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 405_2_17</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Failed</td>
<td>apr 30, 2014 9:17:05 PM</td>
</tr>
<tr>
<td>2.1.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 405_5_17</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 9, 2014 10:11:11 AM</td>
</tr>
<tr>
<td>2.1.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 405_2_17</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 9, 2014 10:17:47 AM</td>
</tr>
<tr>
<td>2.1.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 405_5_17</td>
<td>DE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 9, 2014 11:24:42 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>MA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 9, 2014 1:11:45 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>MA</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Done</td>
<td>mar 12, 2014 1:22:45 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>NZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Deleted</td>
<td>mar 12, 2014 1:22:45 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>NZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Deleted</td>
<td>mar 12, 2014 1:27:55 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>NZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Deleted</td>
<td>mar 12, 2014 1:27:55 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>NZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Deleted</td>
<td>mar 12, 2014 1:27:55 AM</td>
</tr>
<tr>
<td>2.2.0</td>
<td>Rebuilt</td>
<td>Pan Database and VM OPP 410_090_R2</td>
<td>NZ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Deleted</td>
<td>mar 12, 2014 1:27:55 AM</td>
</tr>
</tbody>
</table>
## Appendix F: Example of how the data in Qlikview is presented

<table>
<thead>
<tr>
<th>Month</th>
<th>ELU</th>
<th>EBU</th>
<th>HDD</th>
<th>Buffer</th>
<th>Server</th>
<th>64</th>
<th>32</th>
<th>16</th>
<th>8</th>
<th>4</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan</td>
<td>123</td>
<td>456</td>
<td>789</td>
<td>012</td>
<td>345</td>
<td>67</td>
<td>89</td>
<td>01</td>
<td>23</td>
<td>45</td>
<td>67</td>
<td>89</td>
</tr>
<tr>
<td>Feb</td>
<td>321</td>
<td>654</td>
<td>987</td>
<td>234</td>
<td>567</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
</tr>
<tr>
<td>Mar</td>
<td>456</td>
<td>789</td>
<td>012</td>
<td>345</td>
<td>678</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
<td>12</td>
</tr>
<tr>
<td>Apr</td>
<td>789</td>
<td>012</td>
<td>345</td>
<td>678</td>
<td>901</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
</tr>
<tr>
<td>May</td>
<td>012</td>
<td>345</td>
<td>678</td>
<td>901</td>
<td>123</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
</tr>
<tr>
<td>Jun</td>
<td>345</td>
<td>678</td>
<td>901</td>
<td>123</td>
<td>456</td>
<td>56</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
</tr>
<tr>
<td>Jul</td>
<td>678</td>
<td>901</td>
<td>123</td>
<td>456</td>
<td>789</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
</tr>
<tr>
<td>Aug</td>
<td>901</td>
<td>123</td>
<td>456</td>
<td>789</td>
<td>012</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
<td>12</td>
</tr>
<tr>
<td>Sep</td>
<td>123</td>
<td>456</td>
<td>789</td>
<td>012</td>
<td>345</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
</tr>
<tr>
<td>Oct</td>
<td>456</td>
<td>789</td>
<td>012</td>
<td>345</td>
<td>678</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
</tr>
<tr>
<td>Nov</td>
<td>789</td>
<td>012</td>
<td>345</td>
<td>678</td>
<td>901</td>
<td>56</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
</tr>
<tr>
<td>Dec</td>
<td>012</td>
<td>345</td>
<td>678</td>
<td>901</td>
<td>123</td>
<td>78</td>
<td>90</td>
<td>12</td>
<td>34</td>
<td>56</td>
<td>78</td>
<td>90</td>
</tr>
</tbody>
</table>

### Functions (Example)

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUM</td>
<td>Summation</td>
<td>number</td>
</tr>
<tr>
<td>AVG</td>
<td>Average</td>
<td>number</td>
</tr>
<tr>
<td>MAX</td>
<td>Maximum</td>
<td>number</td>
</tr>
<tr>
<td>MIN</td>
<td>Minimum</td>
<td>number</td>
</tr>
<tr>
<td>COUNT</td>
<td>Count</td>
<td>number</td>
</tr>
<tr>
<td>DISTINCT</td>
<td>Unique</td>
<td>number</td>
</tr>
</tbody>
</table>

### Diagram

[Diagram of Qlikview data presentation]

---

**Note:** The diagram and table are placeholders and should be replaced with actual data and visualizations from the Qlikview environment.
Appendix G: An example of the data found in SDP3 usage

Field test for 2.23

SDP3 2.23 until 2015-04-06
Unique clients: 36

Updated 2015-04-06
Filter SDP3 usage

2014-01-01 to 2015-04-07

Week range: 1 - 15

Major version | Minor version | Revisions | Builds
---|---|---|---
2 | 23 | 0 | Select all
| 22 |
| 21 |
| 20 |
| 19 |
| 18 |

Group by: Week | Day
Include internal clients

Graph dimensions in pixels, default 600 * 400 px

Width px Height px

Set chart title. Default: "SDP3 usage"

Graph title

Draw chart
Appendix H: Summary of data

The three tables below represent a summary of the overall data that has been collected besides the product updates data. The black boxes contain sensible information that due to confidentiality needs to be censored.

<table>
<thead>
<tr>
<th>Version</th>
<th>In-house Testing (Changes)</th>
<th>Field Testing</th>
<th>After Release</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tests</td>
<td>TRs</td>
<td>CRs</td>
</tr>
<tr>
<td>2.18</td>
<td>66</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>2.19</td>
<td>73</td>
<td>85</td>
<td>53</td>
</tr>
<tr>
<td>2.20</td>
<td>253</td>
<td>117</td>
<td>51</td>
</tr>
<tr>
<td>2.21</td>
<td>201</td>
<td>69</td>
<td>35</td>
</tr>
<tr>
<td>2.22</td>
<td>145</td>
<td>66</td>
<td>42</td>
</tr>
</tbody>
</table>
## Appendix I: Correlation values

<table>
<thead>
<tr>
<th></th>
<th>(1)</th>
<th>(2)</th>
<th>(3)</th>
<th>(4)</th>
<th>(5)</th>
<th>(6)</th>
<th>(7)</th>
<th>(8)</th>
<th>(9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Trouble Reports fixed (1)</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of Change Requests implemented (2)</td>
<td>0.311</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of TRs + CRs (3)</td>
<td>0.951</td>
<td>0.589</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of uses during field testing (4)</td>
<td>0.673</td>
<td>-0.29</td>
<td>0.479</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of unique clients (5)</td>
<td>0.962</td>
<td>0.401</td>
<td>0.948</td>
<td>0.643</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of Trouble Reports found during field testing (6)</td>
<td>0.94</td>
<td>0.084</td>
<td>0.827</td>
<td>0.611</td>
<td>0.865</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of issues found after release (FQs) (7)</td>
<td>0.236</td>
<td>0.906</td>
<td>0.495</td>
<td>-0.30</td>
<td>0.223</td>
<td>0.017</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of weeks of field testing (8)</td>
<td>0.92</td>
<td>0.419</td>
<td>0.918</td>
<td>0.693</td>
<td>0.982</td>
<td>0.772</td>
<td>0.235</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Total no. of uses of release candidate and extra release (9)</td>
<td>0.042</td>
<td>-0.83</td>
<td>-0.23</td>
<td>0.694</td>
<td>-0.07</td>
<td>0.122</td>
<td>-0.67</td>
<td>-0.04</td>
<td>1</td>
</tr>
</tbody>
</table>
Appendix J: Distribution of ECU failures

After release 2.20.0

Failure count Cumulative %

After release 2.20.1

Failure count Cumulative %
After release 2.21

Failure count Cumulative %

After release 2.22

Failure count Cumulative %
Appendix K: Restricted framework
The following pictures displays how the restricted framework.

<table>
<thead>
<tr>
<th>Start Version</th>
<th>SDP3 2.18 (SP14022)</th>
</tr>
</thead>
<tbody>
<tr>
<td>End Version</td>
<td>SDP3 2.22 (SP13563)</td>
</tr>
</tbody>
</table>

(Fill in the name and end of the release exactly like it is in JIRA)

<table>
<thead>
<tr>
<th>Date</th>
<th>Release</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013-02-27</td>
<td>SDP3 2.18 (SP14022)</td>
</tr>
<tr>
<td>2014-05-22</td>
<td>SDP3 2.19 (SP14655)</td>
</tr>
<tr>
<td>2014-10-09</td>
<td>SDP3 2.20 (SP1410)</td>
</tr>
<tr>
<td>2014-12-10</td>
<td>SDP3 2.21 (SP1412)</td>
</tr>
<tr>
<td>2015-02-24</td>
<td>SDP3 2.22 (SP13563)</td>
</tr>
<tr>
<td>2015-04-28</td>
<td>SDP3 2.23 (SP13565)</td>
</tr>
<tr>
<td>2015-09-29</td>
<td>SDP3 2.24 (SP13510)</td>
</tr>
<tr>
<td>2015-11-24</td>
<td>SDP3 2.25 (SP13512)</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.26</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.27</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.28</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.29</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.30</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.31</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.32</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.33</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.34</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.35</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.36</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.37</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.38</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.39</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.40</td>
</tr>
<tr>
<td></td>
<td>SDP3 2.41</td>
</tr>
</tbody>
</table>

1. Filter in JIRA with Project as SDP3 and Issue Type as Trouble Reports and Change Requests
2. Export all the data to Excel
3. Delete everything in the JiraData worksheet
4. Copy all the JIRA data to the JiraData worksheet
5. Enter the data in InputData manually
6. Select the start and end version in MainSheet

Note: Ensure that the releases are entered correctly in the table to the left

For ECU Updates
1. Add ECU Updates data at the bottom of the ECU_Updates worksheet
2. Refresh the graphs

Note: Ensure that the releases are entered correctly in the table to the left
<table>
<thead>
<tr>
<th>Version</th>
<th>SDP2 2.18</th>
<th>SDP2 2.19</th>
<th>SDP2 2.20</th>
<th>SDP2 2.21</th>
<th>SDP2 2.22</th>
<th>SDP2 2.23</th>
<th>SDP2 2.24</th>
<th>SDP2 2.25</th>
<th>SDP2 2.26</th>
<th>SDP2 2.27</th>
<th>SDP2 2.28</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of Trouble Reports fixed</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>No. of Change Requests Implemented</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>Total no. of Trouble Reports fixed and Change Requests implemented</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>Total no. of uses during field testing</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>Total no. of unique clients</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>Total no. of weeks of field testing</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>Total no. of uses of release candidate and extra release</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>Total no. of unique clients of release candidate and extra release</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>No. of Trouble Reports found during field testing</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>No. of issues found after release (FRs)</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>No. of uses before release candidate</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>No. of changes during release phase (TRs)</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>Parameter</td>
<td>Comments</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>
</tr>
<tr>
<td>Total no. of uses during field testing</td>
<td>From SDP3 Usage data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of unique clients</td>
<td>From SDP3 Usage data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of weeks of field testing</td>
<td>From SDP3 Usage data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of uses of release candidate</td>
<td>From SDP3 Usage data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of uses of extra release after release candidate</td>
<td>From SDP3 Usage data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of unique clients of release candidate and extra release</td>
<td>From SDP3 Usage data</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of issues found after release (FQs)</td>
<td>From FRAS, refers to the FQs that are new issues (unknown previously) and can be</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of changes during release phase (TRs)</td>
<td>From Jenkins, refers to all the TRs that were fixed in a version after the start of the</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total no. of uses of release candidate and extra release</td>
<td>Calculated automatically taking the uses of the release candidate and, if it exists, the</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Issue Type</td>
<td>Status</td>
<td>Priority</td>
<td>Resolution</td>
<td>Assignee</td>
<td>Reporter</td>
<td>Creator</td>
<td>Create</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>
</tr>
<tr>
<td>Trouble Report</td>
<td>Open</td>
<td>0 - None</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change Request</td>
<td>New</td>
<td>2 - High</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2014-12-6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Change Request</td>
<td>New</td>
<td>0 - None</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2015-06-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Order Type</td>
<td>Client Version</td>
<td>VIN</td>
<td>Serial No</td>
<td>ECU Family</td>
<td>ECU ScmMid</td>
<td>Status Type</td>
<td>Created Date</td>
<td>Veh Type</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>
</tr>
<tr>
<td>Rebuild</td>
<td>2.19.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rebuild</td>
<td>2.19.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rebuild</td>
<td>2.19.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rebuild</td>
<td>2.19.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rebuild</td>
<td>2.19.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software</td>
<td>2.20.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Truck</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

81
Appendix L: Graphs generated in the restricted framework

The following figures display the different graphs that are generated with the restricted framework. The pivot-graphs for the ECU updates have changeable inputs, which means that, e.g. Done ECU updates can be change to Failed.
TRs found during field testing

No. of TRs

SDP3 2.18 (SOP1402)
SDP3 2.19 (SOP1405)
SDP3 2.20 (SOP1410)
SDP3 2.21 (SOP1412)
SDP3 2.22 (SOP1503)

Version

No. of Trouble Reports found during field testing

Issues found after release per version (FQs)

No. of Issues

SDP3 2.18 (SOP1402)
SDP3 2.19 (SOP1405)
SDP3 2.20 (SOP1410)
SDP3 2.21 (SOP1412)
SDP3 2.22 (SOP1503)

Version

No. of issues found after release (FQs)

No. of changes during release phase (TRs)
Issues found after release per version (FQs)

No. of Issues

SDP3 2.18 (SOP1402)  SDP3 2.19 (SOP1405)  SDP3 2.20 (SOP1410)  SDP3 2.21 (SOP1412)  SDP3 2.22 (SOP1503)

No. of CRs

Version

No. of issues found after release (FQs)  No. of Change Requests implemented