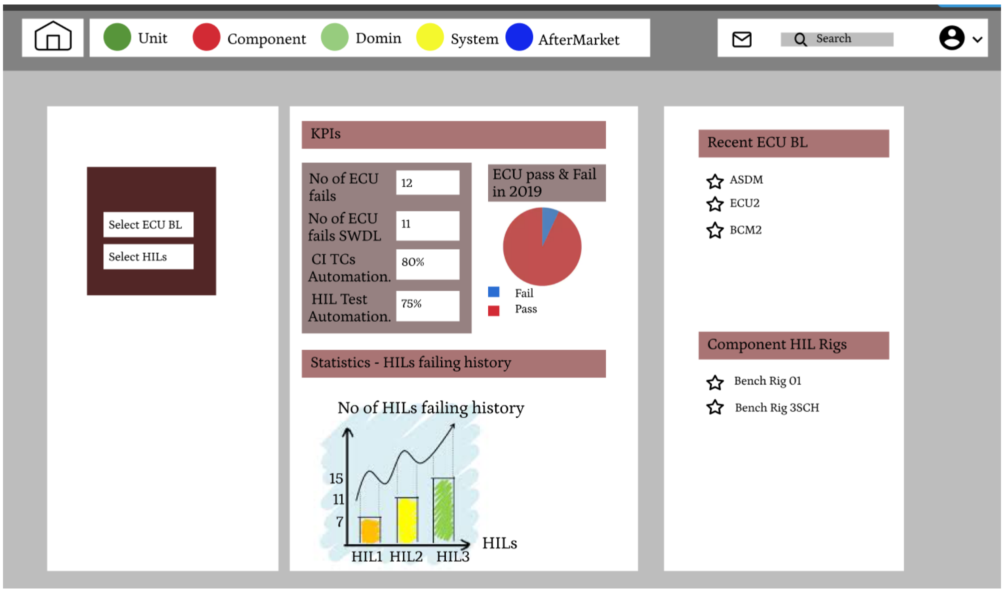


How to gain a quality overview in Complex large system:

An exploratory case study of self-driving car project at Volvo Cars.

Master's thesis in Software Engineering

Kalayu Yemane Berhe
Rayna Singh Sengar

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2019

Master's thesis 2019

A thesis report on exploring a quality overview solution for a complex domain, using an exploratory case study, where quality overview is explored during the continuous development of a complex system in the case of autonomous drive systems (car project).

Kalayu Yemane Berhe and Rayna Singh Sengar

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2019

A thesis report on exploring a quality overview solution for a complex domain, using an exploratory case study, where quality overview is explored during the continuous development of a complex system in the case of autonomous drive systems (car project).

KALAYU YEMANE BERHE and RAYNA SINGH SENGAR

Supervisor: Eric Knauss, Department of Computer Science and Engineering
Advisor: Erik Sternerson, Volvo Cars
Examiner: Regina Hebig, Department of Computer Science and Engineering

Master's Thesis 2019
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: The picture on the cover page is a quality overview alternative solution prototype, that has been evaluated by expert users/stakeholders in the domain, mapped from the qualitative model which is the final output of the study.

Gothenburg, Sweden 2019

Quality Overview in Complex Domain: An Exploratory Case Study of an Autonomous Car Project at Volvo Cars.

KALAYU YEMANE BERHE and RAYNA SINGH SENGAR
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Tracking of quality state of products during continuous development is important, especially if the development is carried out in a complex domain. Perhaps, a good quality overview would enable effective decision making of stakeholders on progress status, follow-up tasks, and improvement plan. In practice, while many stakeholders might be experts in their scope of working and might have a deep understanding of the quality state of their part of the product, reviewing the quality state of products at an integration level or end-to-end is quite challenging as most stakeholders undergo the same scope. Although visualizing verification activities (testing strategies) has been addressed in previous studies, there is a need to study how to deliver a quality overview at an abstraction level that would give a convenient way of presenting the quality state of products to every stakeholder. In this study, we explored this gap and reported on an empirical study that investigates challenges, opportunities of the current quality overview and possible elicitation of quality overview attributes in aspect with different scopes from the different stakeholders. In our exploratory case study, we have conducted semi-structured interviews with 16 stakeholders. As a result, we have explored the chain of challenges in the foundation, life-cycle, and visualization of quality overview, together with valuable opportunities of certain quality overview tools which are existing in the case company, identified the convenient key performance indicators(KPIs) and how different stakeholders prefer to get their quality overview visualized. In the end, we proposed an alternative quality overview solution that appeared to be gone through an expert review evaluation in the case company.

Keywords: quality overview, KPIs, complex domain, visualization, testing, requirement, integration level, stakeholders, verification, prototype.

Acknowledgements

Our first acknowledgment goes to our academic supervisor Eric Knauss for guiding us throughout the study. We would like to thank our examiner Regina Hebig for the constructive feedback and follow-ups, as well. Furthermore, we would like to extend our special acknowledgments to Volvo Cars and the company supervisors, Tony Heimdahl, Erik Sternerson, and Piret Saar. The hospitality and support were vital during our work. In addition, we want to thank the Swedish Institute (SI) Scholarship which greatly supported the studies for one of the authors (Berhe). Last but not least, we would like to thank our friends and families for their support and encouragement.

Berhe and Sengar, Gothenburg, August 2020

Contents

1 Introduction

Good quality has always been an important feature in the area of transportation because it ensures safety on the road [9]. Nowadays the vehicle industry is turning towards the autonomous domain where most of the actions of the vehicle are controlled by the vehicle itself. Therefore tracking down the quality in its development process every now and then has become a necessity. Our study emphasizes the importance of a good quality overview at each level of the continuous development of complex systems in the case of autonomous drive systems.

This study aimed to create a quality overview model, that eventually turned in to a quality overview solution, in the continuous complex development of the autonomous domain. Therefore, in this report, the continuous development, complex systems, and autonomous domain are the three main areas to deal with. All these three areas are interrelated with each other which is described more briefly in the next paragraphs.

An autonomous drive enabled vehicle is a highly complex system which consists of various components (i.e. electronic control units or sensors). These components have an order of software modules [10], embedded in it and the logical components are cross-cutting combined in different ways to create the complete system which in a way is very complex. With the increasing effort need to provide, advanced features and safety concerns, the growth of the software and electronics in an autonomous drive enabled car is exponentially increasing and so is the demand for quality in it. The development process of such autonomous drive enabled vehicles mostly follows a continuous development process consisting of continuous integration, verification, and validation in an agile framework which is a key enabler for meeting the increasing demands on short development cycles.

Several development organizations have already implemented continuous integration (CI) flows in place, with automated verification and validation, but in scenarios where the product is very complex, (e.g. autonomous drive vehicle that has a hundred of electronic components and thousands of software modules, and hundreds of variants of the complete product) understanding the quality of the product and present it to respective stakeholders in a suitable quality overview visualization is challenging.

Furthermore, organizations that utilize large-scale agile development methodology require understanding the quality of their product for guiding the development efforts. These organizations need quick and accurate feedback on the quality (or a quality overview) of products, in a format that supports developers, project managers, and other stakeholders to grasp the information quickly for a quick and reliable decision making. Thus, this scenario shows an exploratory case study in such area might be needed. Agile development is an iterative process that puts the focus on working software instead of early and rigid requirements specification, with work carried out in empowered cross-functional teams with continuous customer input [8]. CI, validation, and verification play a crucial role in it.

In this study, we explored the challenges, opportunities, key performance indicators, and visualization methods of the current quality overview that support stakeholders during the elicitation of the quality state of their products. By conducting an exploratory case study with 16 stakeholders, we have discovered the challenges and opportunities of current quality overview in the stakeholders' perspective. Furthermore, we have uncovered the valuable key performance indicators and convenient visualization preferences that are chosen by the participants of the study. Based on the findings, we have developed a qualitative model that has a summary of the chain of challenges, opportunities, and quality overview attributes which is eventually mapped into a quality overview solution via a prototype.

The main contributions of this thesis study are presented as follows:

- We investigate and identify the chain of challenges of the current ways of presenting of quality state of products using our case study. These provide value when reasoning about how to improve them and propose an alternative solution.
  Example: Some of the challenges investigated by the study are:
  - A well-organized quality overview in each scope and end-to-end that would incorporate manual verification activities is lacking.
  - Easy way of collaboration between/among stakeholders and project levels is lacking. (Or, communication among stakeholders is mostly manual-based. Example: Excel sheet report)
  - Visualization of/for categorization of failure is missing. When failure issue (of products) appears, there is no way if the failure is due to the product under test, testing environment setting or the other product integrated with.)

- We list potentially valuable key performance indicators from our case study. These provide value when reasoning about which information to include in a quality overview and in this way supports both research and industrial practices.
  Example: Some of the KPIs identified in the study are mentioned below:
  - Requirement coverage.
  - Test case automation percentage.
  - Problematic features/test cases percentage.

- We investigate and identify convenient visualization techniques during product

development from our case study. These provide value when reasoning about how to visualize quality overview.
Example: Some of the visualization preferences elicited from the interviewee are mentioned below:
  - Customize-able visualization dashboard that can be applied for all project levels.
  - Interactive dashboard with an option of drilling down the depth information about the quality state of products.
  - An optimal single visualization platform connecting all other existing quality overview tools.

- Propose an alternative quality overview solution, that can be considered as one of the possible solutions, which tries to incorporate the quality overview key performance indicators, stakeholders' visualization preferences, opportunities of the existing quality overview and the challenges investigated during the study.

This study is an exploratory case study based on semi-structured interviews with participants from the case company. We suggest some fields in this domain for future work to investigate this topic more exhaustively. For example, we would suggest future studies give emphasis on key performance indicators and its possible negative influence on the mind of developers. Also, in this study, understanding the concept of quality overview in-depth within this case was prioritized over the generalized results. However, we suggest future studies to incorporate the view from different cases as well. Furthermore, we didn't implement a quality overview solution tool, instead, we developed an alternative solution prototype tool that is mapped from the qualitative model found from the study and we are confident that this prototype would prove to be valuable when implementing tools support for quality overview in similar contexts. We have also tried to incorporate all findings from the challenges, opportunities, critical KPIs, and visualization preferences as per the view of our interviewees.

1.1 Purpose of the Study

The purpose of the study is to develop a qualitative model, as a foundation for the alternative solution prototype, that summarizes the challenges and opportunities of current ways of delivering products' quality state overview information in an end-to-end Car project. It has also aimed to identify the valuable key performance indicators that have to be elicited from the result of verification activities on the various products at an integration level. Finally, it aimed to explore stakeholders' preferences in the aspect related to their choice of quality state visualization techniques.

The finding from the qualitative model is expected to be used as an important input during creating an appropriate quality overview solution in general and as a foundation of the proposed alternative solution prototype in this study in particular. The proposed alternative solution prototype display quality overview information along with key performance indicators for the various stakeholders at an appropriate level of detail.

1.2 Research Questions

Given a complexity and huge amount of combinations in the domain, creating and maintaining a quality overview is challenging. Thus, a proper study in the domain is important. An exploratory approach is, therefore, a convenient approach to study such a domain with complex behavior and level of detail. However, before developing a tool that is supposed to deliver an appropriate quality overview, the domain shall be explored and the challenges should be identified.

Previous literature has studied an end-to-end concept model that could visualize the different verification activities in integration levels [3], there is however less research that investigates the quality overview in an abstraction level that would be convenient to most stakeholders. For that purpose, therefore, we have, through our study, investigated the challenges, opportunities, and quality overview attributes of products at an abstraction level.

Hence, we have specified the following research questions and its description below:

RQ1 - What are the opportunities and challenges of the current quality overview?

This question aimed to investigate the opportunities and challenges of the current ways of delivering information about quality state products across different project levels. This question serves as a base for the rest two research questions.

RQ1.1 - What are the challenges of the current quality overview?

This question aimed to uncover the current quality overview present in each level of the domain which leads to the challenges faced by the different stakeholders and areas to improve in the current quality overview. This question will also help in making the stakeholders aware of what they should expect from the other stakeholders which could increase the transparency between each level by uncovering some hidden aspects of the current quality state overview.

RQ1.2 - What are the opportunities of the current quality overview?

This question aimed to discover the opportunities of the current quality overview tools and ways of conveying product quality state information in the domain. Different opportunities are explored.

RQ2 - What are the Key Performance Indicators (KPI) desired by stakeholders?

This question aimed to identify what kind of KPIs are needed by the stakeholders at their scope (and in an integration level) to easily grasp the quality information or status. And, what KPIs they would like to see in other levels for their understanding and make necessary decisions.

RQ3 - In what ways would the stakeholders prefer a quality overview to get visualized?

These questions aimed to find a good visualization for the quality overview preferred by various stakeholders at their domain i.e. how they want to view the quality status of their product or for the other levels in an easy way. This question also highlights the present visualization dashboards for the verification activities happen at each level and its contribution to judging the quality of the product by the various stakeholders.

1.3 Outline

This thesis report is structured as follows: Chapter 2, we primarily present background which consists of terminologies used throughout the study. It also covers the different levels of the Car project along with possible verification activities. Chapter 3 presents the related work and literature's results in connection with our study. The research methodologies applied can be found in Chapter 4. A representation of the gathered results and the proposed solution along with its evaluations are presented in Chapter 5. The results are then discussed concerning the research questions in Chapter 6. The threats to validity during the study and future works are also discussed in this chapter. Finally, this study is concluded by a summary of the thesis work in Chapter 7, a conclusion.

2
Background

This chapter clarifies the concepts and terminologies used throughout this study. It also covers the information regarding to the case company, the domain under study, and the different levels of the self-driving car project along with its corresponding verification activities and existing quality overview tools.

2.1 Case Company

The case study is conducted in a case company called Volvo Cars Corporation. As one of the most well-known and respected premium car brands [1], the company is thriving towards the development of autonomous cars (self-driving cars) in their Active Safety and Autonomous drive domain where we appeared to conduct our study at.

As the domain undergoes through the development of a complex combination of components, the company currently adapting the agile way of working called SAFe. During this continuous development of products, continuous verification and validation are relevant to ensure the quality of products. Above all, how to present the quality overview of verification and validation activities to various stakeholders of different scopes across product development is challenging, where the need for our study appeared. Despite the fact that our study is limited to a single case study, understanding the concept of quality overview in-depth and propose an alternative solution within this case was prioritized over the generalizability of the results.

2.2 Autonomous Drive Domain: Self-driving car project

Autonomous drive domain is the domain under study in the case company. Hence, We explored the existing verification activities across the levels of the products development in this specific domain.

As per verification and validation of the self-driving car project, the domain roughly has four product levels which reflect the software integration complexity on which the verification carried out. Those are a unit, component, domain, and system product levels. Each level has inputs such as test objects, test cases, and analysis scripts.

[1] https://group.volvocars.com/company

And, they have a test environment such as virtual environment, test benches, HIL rigs, Boxcar, Complete HIL, and Real car.

2.2.1 Self-driving Car Project Development

The self-driving car project has roughly three types of baselines[2] across the product levels. The baselines are component (ECU[3]) baseline, domain baseline, and system baseline. There is no baseline for unit product level, as the first baseline in the project is build as a collection of software delivered by suppliers or in-house. Hence, the first baseline is component baseline for a particular ECU with a collection of software modules as a VBF[4] file format.

Baselines and product levels play an important role in the quality overview of verification and validation activities. Baselines are the collection of items that are bundled together. For example in our case. if we take Component/ECU Baselines, in order to make a baseline for one ECU, it takes a collection of VBF files and its corresponding meta-data. Whereas, product levels are the categories which show which baseline is to be verified at which product level. Product levels are set according to the industry standard. Baseline testing captures the performance information whereas verification according to the levels shows application performance within the scope of that particular level. The three baselines are described as follow:

1. **Component/ECU Baselines -** are a collection of VBF files for a particular ECU and its corresponding meta-data. As per verification, the ECU baselines are tested on Component rigs.

2. **Domain Baseline -** is a collection of component baselines and its corresponding meta-data that creates a complete domain. Domains are built as a collection of ECUs, and ECUs are built as an integration of software modules. For Example, Active Safety, IHU, etc. As per verification, the domain baselines are tested on Domain HIL. Domain HIL is a Hardware In Loop rig that contains ECUs for a particular domain. It used for verifying domain baselines that contain software for domain ECUs.

3. **System Baseline -** System baseline is a collection of Domain baselines and its corresponding meta-data.

[2]Baseline - is a collection of items, in our case VBF files and meta-data, that are bundled together and considered as one entity in further handling.

[3]ECU - Electronic Control Unit: ECUs are the hardware units in the car, that together with the downloaded software make up the intelligence of the car.

[4]VBF - Versatile Binary Format: Package format that adds a header before the software itself, that tells the software download process what kind of software it is and where to put it. VBFs are the software modules for the ECUs.

2.2.2 Self-driving Car Project Verification Levels

There are four project verification levels which comprise its own verification inputs and testing environment settings/configurations[5].

1. **Unit** - is a level where verification of software modules, that supposed to be integrated into an ECUs, is performed. The inputs are software modules as test objects, automated/ non-automated test cases and analysis scripts. The test cases are run in a virtual environment to make sure the software is working as it intended to be.

2. **Component** - is a level where an integration of software modules, that are integrated into a particular ECU/sensor, is tested. The inputs are VBF files (or Software compositions) as test objects, ECU level test cases, and analysis scripts. The ECU level test cases are run in a test benches/rigs.

3. **Domain** - is a verification level where the integration of software modules that are integrated into an ECUs of the same function or domain is tested. Hence, a function/domain is tested against the domain requirements at this level. The inputs are ECU baselines (VBF files) as test objects, domain-level test cases, and analysis scripts. The domain level test cases are run in Domain HIL rig.

4. **System** - is a verification level where an integration of software modules that corresponds to the whole system/whole car is tested. The entire functions of the self-driving car inline with the complexity of the integration of software modules across the baselines beneath to this level are tested. The inputs are domain (VBF files) as test objects, system-level test cases, complete car level test cases, and analysis scripts. In this level, the test cases are run in two types of testing environments, those are Boxcar and complete cars. In the first testing environment, the intensive base-tech requirement testing is done whereas in the latter the complete electrical system testing is done.

2.2.3 Verification and Validation Activities

This section introduces the verification activities happening across the product levels in the domain understudy of the case company. After the domain exploration during the study and the research findings, the verification activities explored are presented as follow:

- **Acceptance software testing**

  This kind of testing is done at a unit level, also referred to as the pre-ECU level. Stakeholders on this level run unit tests for software modules delivered

[5] Test Environment Configuration - is a set of properties that describes the hardware and software part of a test environment, and how they relate to each other.

or developed in-house. They make sure test cases meet the required acceptance criteria and the requirement coverage demanded.

In the case company, after software modules are delivered in a VBF file format from suppliers, acceptance testing is conducted by the stakeholders to check if the software implementation is functioning as it should be prior further integration with the other software components to create an ECU baseline. Then, an acceptance test reported is generated.

- **Virtual car simulation testing**

  This kind of testing is of a software (unit level) testing, run in virtual test environment settings. The virtual car takes software, having not to worry about the hardware and other software, and make the simulation of testing the logic of the software. It is placed in the continuous integration (CI) chain where the software is run with different automated scenarios.

- **ECU/Component testing**

  In this testing, ECU's verification is done on the component testing environment setting, component test rigs. In order to test a particular ECU, other ECUs are simulated on the test rigs which is an open loop.

- **Function testing**

  This kind of testing is a type of verification in domain level where the functional, safety, and legal requirements of domains are confirmed. All domains should go through both functional and legal tests before it goes to a complete car. The functional testing is run in Domain HIL[6] rigs. In this environment settings, the Domain HIL developers create a model and scenarios, where they simulate certain parts that are needed for the testing.

  There are various domains that exist in the case company, like Advanced Driver Assistance System, Active Safety, Infotainment Head Unit, etc. where the integration of software modules in multiple ECUs form the various functions across domains. So, the main aim of the testing of the function is to make sure if a particular function works properly and complies with the pre-specified requirements.

- **Base-tech testing**

  This verification type is done in system or complete car verification level where base-tech test cases run in Boxcar testing environment settings. The test cases related to the boxcar testings are comprising of software download, gateway,

[6]Domain HILs - Domain HIL is a Hardware In Loop rig that contains ECUs for a domain. Domain HILs are used for verifying domain baselines that contain software for domain ECUs.

network management, and diagnostics. After this testing is done, developers or test experts will be able to determine the basic connectivity in the complete car as a backbone for the application testing in the lower levels such as unit, component, and domain.

- **Complete HIL testing**

  Complete HIL is a kind of environment setting that falls in the system level and comes after boxcar testing. In this verification phase, the driver scenario, driver's speed, etc. are simulated along with the validated functions. Application testing and base tech testing both can be performed in complete HIL testing.

- **Real Car testing**

  This testing type is a kind of testing on an actual car. This can be done by driving cars so that to measure the function's rating requirements or after-market performance. The testing of actual cars against a set of scenarios makes a valuable contribution to make the cars ready enough before going into the market. Besides, the after-market performance of cars, software downloading failure rate, for instance, makes a valuable contribution to do a quick resolution and give information about the number of issues opened during the aftermarket.

2.2.4 Current Quality Overview Tools

In the self-driving car project development and its corresponding verification activities, stakeholders have some existing ways of conveying the result of the verification to users and other stakeholders in their domain or elsewhere. As per the domain exploration during our study and the data collected from the interview session, we have come to uncover some ways of presenting a quality overview of the different project levels.

As can be shown below, Table 2.1, we have summarized the explored current ways of providing quality state of products categorized as events, tools, and communication channels.

1. **Events/Meetings**

   Events or meet-ups are one of the current ways of conveying quality information of products in the domain under study. For example, Agile events and quality information presentations. During these events, stakeholders discuss the current quality state of products across the levels of the project.

   Agile events

   Through these events team members of each team present their work to the

Quality Overview Tools			
No	**Quality Overview Approach**	**Tool**	**Level (Scope)**
1	Events/Meetings	Agile events and Presentations	Agile meet-ups in all levels and Presentations in domain level.
2	Tools	Local CI flow board, JIRA/VIRA, System Weaver, Reports, Power BI and Grafana	All tools for all levels (System Weaver for unit and component)
3	Communication	Sharepoint, Reports	All levels.

Table 2.1: Quality overview tools in the self-driving car verification of the case company

whole department or within themselves. This helps them in determining the speed and performance of their progress and its alignment with the quality strategy. According to the data collected in the study, agile events are one of the existing ways of conveying quality overviews which appears to happen at different times, such as daily scrum, sprint planning, and demo. Stakeholders discuss their progress and present quality information.

Therefore, this is one of the ways to get quality coverage as per teams by looking into the number of test cases executed, test cases pass/fail via graphs and some other illustrations.

Presentation about quality state of functions

In the case company, every release train presents product development status or quality status every week or second week specifically to their solution management. This is the main existing quality process right now at the solution level.

2. **Currently available tools**

Apart from the events approach, as shown in Table 2.1, several tools exist that are used on all levels. Following are some descriptions of the tools:

Team-wise CI flow board

Local CI flow, in all teams of the case company, is one of the tools that stakeholders use as a way of showing product quality information periodically. Many teams, in the domain under study, have a CI flow board which shows how many test cases have been passed or failed. One can determine the number of faults by looking into the quality information in the CI flow board. Stakeholders from any level also can view the status of their product by looking into that flow board.

Even though CI flow board shows the quality of the product on the test case level which is convenient for the lower levels, when it comes to tracking of the quality of product at higher levels, it is hard for the stakeholders to understand the quality overview information easily because of the absence of the quality information in an abstraction level, such as feature readiness.

JIRA (Customized as "VIRA" in the domain under study)

Stakeholders use this tool for a failure issue tracking purpose which is present in all the project levels, where a ticket is being created by stakeholders with a more brief description along with reproduction steps, actual and the expected test results. Based on the short description in the ticket, some other stakeholders or the same person would update the issues and shows the issue status instantly. Hence, through this tool, stakeholders could track the status of issues and resolution time. In this way, this is being used as one of a quality overview tool.

System Weaver - Requirement handling tool

Stakeholders in unit and component product levels use the tool to convey quality information regarding software modules and the corresponding integration. It is a tool where all the requirements, test methods, and test results are recorded and tracked according to the product specification. Stakeholders use this tool as a quality overview tool because it traces the requirements by showing the test progress, test case mapping, and test result.

Power BI - Visualization tool

Stakeholders use this external tool by Microsoft, which is currently being used as one of the current ways of presenting quality overviews to collect data from all other systems and get visualized on a web browser. For example: To track continuous progress in CI, Power BI is used to visualize the number of software being delivered into the CI repositories.

Grafana

Grafana contributes to the quality overview with a focus on developers on any level, by providing test failure statistics. This allows the stakeholders to assess test case stability.

3. **Communication channels**

As it can be shown in the table, Table 2.1, along with the event and tools as the quality overview approach, Stakeholders also do sort of communication channels among themselves as a way of sharing quality overview information in and across the project levels. Those are in the form of reports and an external tool called Sharepoint.

Test Reports

This approach is available in all levels in the form of an excel sheet report, stating the status of the product. At the unit product level, suppliers/in-house developers deliver the test report for their software modules which mainly states the status of the software verification results and the corresponding requirements' fulfillment. Then, the stakeholders at the unit product level, in the case company, do the acceptance testing of the software delivered by the suppliers/in-house developer and send the acceptance test-result's report back to the suppliers/in-house developer in the form of an excel sheet, showing if the requirements meet the needs. In this way, excel sheets reports are being used back and forth at the unit level. This way, stakeholders used this approach to present a quality overview, although it has limitations that can be described in the section "Research Finding" of this report.

In the test environments, reports are being used as a source of product quality information of the product under test. Hence, all the verification results and information are reported in an excel sheet. The stakeholders in these environment settings put the status of their environment settings/HIL rigs in an excel sheet through which HIL developers or testers get the information regarding the quality state for HIL. In other words, they get to know which ECU, car or models are supported. Post verification and test results are also recorded in the excel sheet.

Furthermore, in the base-tech testing, at the box-car level, the testing results of all levels are generated in the form of excel-sheet report. Hence, this is how the reports serve as an important quality overview tool in base-tech testing.

Sharepoint

Many stakeholders also use this tool to discover products and their corresponding testing verdict and product details. For example: If an issue is found while testing an ECU against a base-tech test, stakeholders use the Sharepoint tool as a platform to find out information regarding this ECU, such as who is responsible, testing result, and overall detail. In this way, this tool is being used as one of the tools that show quality information of products.

Finally, as per the exploration of verification activities and existing quality overview tools in the domain, we came to summarize the project levels, testing environments, verification activities, and its corresponding current ways of delivering quality state status information as it can be shown in the table below, Table 2.2. All project levels have some common current quality overview tools[7] on top of the existing tools specific to their level.

[7]All common quality overview tools - are the tools that are explored during the study which happened to be common on most of the project levels. Those are agile events, team-wise CI flow board, Jira/Vira, visualization of test reports using external tools, communicating with some stakeholders for issues regarding verification and VCC CI visualization.

No	Integration level	Verification Activities	Current Quality Overview Tools	Test Environment
1	Unit	Acceptance software testing	All common current quality overview tools + System Weaver, Excel acceptance test report and Excel software test report by suppliers during delivery	Virual Cars Simulation Settings
2	Component	ECU testing	All common current quality overview tools + System Weaver, Excel sheet acceptance test report and Excel sheet ECU test report by suppliers.	Component test rigs environment settings
3	Domain	Function/ domain testing	All common current quality overview tools + Product quality presentations and Excel sheet HIL test report	Domain HIL rigs testing environment settings
4	System-Boxcar	Base-tech testing	All common current quality overview tools + Excel sheet Base-tech test report	Boxcar testing environment settings
5	System - Complete Car	Complete electrical testing	All common current quality overview tools + Excel sheet report	Complete HIL

Table 2.2: Project levels/Integration level, verification activities and its corresponding current quality overview in the Autonomous Drive Domain.

2.3 End-to-end quality overview

In this section, we present the existing quality overview, the need and quality overview solution in aspect to conveying product quality information in an end-to-end product development perspective.

2.3.1 CI Visualization - currently available tool

The case company uses a tool entitled VCC CI Visualization which is a web-based tool that displays the status of the local CIs with the purpose of providing the developers with a way of following their commit through the CI machinery to be able to see the consequences of their change. This tool provides an end-to-end view perspective of the project. However, it does not provide transparency and a comprehensive view of the status of product development from an end-to-end perspective for all stakeholders.

Despite the fact that this tool provides the tracking of quality of the project as it goes through the different levels, the challenges go to when stakeholders are in a need of quality overview information at an abstraction level for the purpose of decision making and improvement plans. This is one of the reasons for us to conduct the study in this domain.

2.3.2 Quality Overview Solution

In this section, we present the need and review of quality overview solutions in complex domain, like the self-driving domain, the domain under study.

The need of end-to-end quality overview solution

Quality has an important role in developing complex systems, such as in an automotive domain. Good quality overview enables requirements traceability and early quality analysis of a system [1]. Continuous end-to-end quality assurance, along with proper ways of presenting and visualizing it to stakeholders as per their level of understanding and detail, is important for continuous development, to deliver a verified system. Thus, keeping track of the quality overview in an efficient way and providing it for the respective stakeholders in a different format is equally important, since the domain might have several levels along with stakeholders who would require a good quality overview in a different level of detail.

However, the complexity of an autonomous system, also with the aim of continuous delivery enables many issues. One such issue is the difficulties of ensuring adequate performance and confidence throughout the life of a product, and thus it is challenging to maintain a bigger picture, at an abstraction level, of its quality state overview.

The visualization of an end-to-end quality overview of a product in continuous development is important for many organizations, especially to organizations that handle complex systems. With an end-to-end quality overview, we refer to all quality overview during verification, from unit testing written by individual engineers to product release. Currently, visualizations and interaction techniques are applied to obtain better insight into and control over the continuous development with an end-to-end verification. Even though those visualization methods are encouraging, it would be optimal to maintain the visualization with a more stakeholder specific overview along with a desired key performance indicator. Thus, it is also crucial to maintain an overview of the quality information with regards to an appropriate level of details (unit, component, domain, and system product levels) in a suitable format to the various stakeholders. This supports the stakeholders, in recognizing and finding relevant information more easily.

Furthermore, [17] software development companies are increasingly striving towards continuous integration, agile development practice, in their efforts to deliver high-quality software faster and faster. The agile movement advocates flexibility, efficiency, and speed to meet the ever-changing customer requirements and market needs. However, in an agile context, figuring out the suitable requirements in each stakeholder level and a system level is still challenging, which could be categorized as [14], a shared understanding of user value (common understanding among stakeholders) and building and maintaining system understanding (system-level view), thus maintaining a quality overview, in such manner, would have an important role to track the progress and releases. Thus, stakeholders would get a quick solution to the quality concern, and that ultimately helps businesses confidently release their products into the marketplace. For Example: will this be a high-quality release? or which parts present the most risk?

In conclusion, having the complexity, amount of combinations, and releases in the autonomous drive domain, a quality overview information with desired KPIs, during the end-to-end verification, to different stakeholders, on different levels and in a different format is still in need. Thus, with the aim of maintaining the appropriate quality overview visualization, we aspired to investigate the current quality overview visualization, the challenges, opportunities, and the stakeholder-desired key performance indicators (KPIs).

A literature review of end-to-end quality overview solutions

One of the solutions that explored through the survey of the literature is the CIViT model [18], a model with an aim to visualize the testing activities that an organization deploys to achieve the desired quality levels during the development of a product or system in an end-to-end product development perspective. The model helps to gain a clear overview and understanding of the end-to-end process of testing activities in a complex domain, in spite of its limitation to give the whole overview enriched with key performance indicators and visualization preferences. This model gives an end-to-end visualization of testing activities across the different verification levels which could be important for any stakeholder to get informed regarding what kind of testing activities are being held at which level. This can also encourage the stakeholders in upper management to get an insight on which verification level needs improvement plans when it comes to testing activities and strategies. However, the tool doesn't show the quality overview, the meaning to testing results, which is accompanied by key performance indicators and stakeholders' visualization preferences, in each verification level which happened to be the main interest of this study.

3
Related Works

This chapter presents related work in the context of delivering quality state of products in complex domain.

3.1 Visualizing Testing Activities and Status

In the work from Nilsson and the other co-authors [3], a suitable visualization technique is studied with the focus of visualizing an applied testing activity explicit to all involved stakeholders. This aimed at creating the basis for the communication between these different parties/stakeholders. In their multiple case study [3], the researchers investigated the lack of an adequate overview of companies' end-to-end testing activities, which tend to lead to problems such as double work, slow feedback loops, too many issues found during post-development, disconnected organizations, and unpredictable releases schedules.

While exploring the testing arrangements at five different software development sites, via the case companies, the researcher studied the challenges regarding the visualization of testing activities in an end-to-end continuous integration pace. Based on the findings, they developed a holistic Continuous Integration Visualization Technique (CIViT) model to provide a useful overview of end-to-end testing activities. However, in contrast to our study, a specific visualization technique (stakeholder-specific), enriched with quality attributes (such as key performance indicators and visualization preferences), that is applied for a different verification level with respect to the detail of quality overview is not explored. For example, stakeholders on upper management or combination responsible would choose to acquire suitable and valuable key performance indicators as a quality overview visualization than an overview of how testing activities are arranged in each verification level.

The CIViT model is aimed to visualize the testing activities that an organization deploys to achieve the desired quality levels during the development of a product or system [18]. The important contribution of the studies [18, 3], is to address the lack of holistic and end-to-end understanding of the test activities for different stakeholders. This was a valuable input to our study, as we have the 'overview of the end-to-end testing activities', in common, that could be applied for the purpose of creating improvement plans or quick decision making. However, the researchers, in the literature[3], seems to focus with a limited number of testing types.

Those are [3]:
- New functionality
- Legacy functionality
- Quality attributes
- Edge cases

Moreover, Stolberg [36] provides a way to visualize the status quo of the quality assurance process before and after implementing a continuous integration environment. Thus, his work focuses on the entire tests and delivery process using an experience report. As per his results, continuous integration along with the V&V improved quality assurance, however in contrast to our study, he did not assess an end-to-end quality overview visualization across the continuous development. Hence, the literature has an impact on the quality assurance of continuous development but with a narrow scope and approaching complex systems is missing.

Besides, unlike the above two studies [18, 3], we were concerned with maintaining a good quality overview with visualizations of every testing strategy including manual verification, in addition to the exploration of quality attributes, KPIs and visualization preferences, on each verification level.

3.2 Quality Requirements & Assurance Challenges

In the work from [37], empirical evidence on the challenges that agile practitioners in organizations currently face when dealing with quality requirements in Agile. After they conducted qualitative exploratory multiple cases, in the context of real-life large-scale distributed Agile projects, the researchers uncovered 13 quality requirements challenges that are classified into five categories.

The categories of challenges identified by the study [37]:

- Team coordination and communication challenges - consists of challenges related to the sharing of information resources among the teams within a large Agile project team.
- Quality assurance challenges - consists of challenges related to inadequate quality requirements test specification, simulated integration tests, and end-user acceptance of quality requirements.
- Quality requirements elicitation challenges - These challenges refer to different aspects of identifying the right quality requirements from the right stakeholders.
- Conceptual challenges of quality requirements - These challenges refer to the conceptualization of quality requirements by the Agile teams.
- Software architecture challenges - These challenges are concerned with linkages between the architecture processes that usually happen in a large project and the Agile development processes, and the architect's role and the Agile roles.

The main contributions of the study [37] are the explication of the challenges from practitioners' perspective which happened to be similar to the first part (first research question) of our case study. The results from both studies helped the practitioners in understanding the challenges and the alternative ways of doing it by identifying the critically of the challenges. Furthermore, a quality requirements management model is developed by [23], as support for handling decisions about quality requirements as a strategy and operational level, encompassing product decisions as well as business intelligence and usage data.

3.3 Visualizing Quality Overview in Complex Domain

A literature [27] presents a quality improvement through visualization of software and systems. They showed that tightly coupling the specification and selection of visualization techniques with the specification and selection of metrics used to collect measurement data better supports the visual recognition of software products, process and project characteristics, and their interrelations.

The important contribution of this literature, with respect to our study, is how they dealt with highly complex structures and nesting. They presented a visualization technique aimed at a better understanding of system properties such as safety and security in highly complex, dynamic embedded systems [27]. As per our study, we have aimed to achieve a quality overview solution for a complex system under continuous development and delivery (releases). Hence, while their focus on how to visualize complex systems to enable software and system understanding is important, a continuous overview in such a continuous system development is missing.

3.4 Implementing Key Performance Indicators

Key Performance Indicators (KPIs) are qualitative and quantitative measures used to review an organization's progress against its goals [34]. It has an important role to play in maintaining the quality overview of products and systems at large. Before identifying the KPIs desired by the stakeholders at different levels of the case company, the study of the different ways to implement KPIs was necessary for the researchers.

In the work from [21], a framework is developed for the organization to evaluate and improve their KPIs to be able to measure their business performance in a more effective way. This study also presents information regarding scientific theories about what good KPIs are and how to develop successful KPIs in an organization where the characteristics of the good KPIs are mentioned and those are aligned, owned, predictive, actionable, few in number, easy to understand, balanced and linked, trigger changes, standardized, context-driven, reinforced with incentives and relevant. Out of these characteristics, few were useful and supported us to come up with the valuable KPIs for the quality overview at different product levels. The proposed framework of this study identified several factors regarding how to develop successful KPIs which include the effectiveness of KPIs, the alignment to business vision and strategies, and the completeness of KPIs. This framework was developed while combining all the relevant theories. By putting all the critical values in one framework, the author believed, it would be an effective tool to help organizations evaluate their strong and weak areas of their KPIs and to improve them constantly. This framework was implemented successfully in one of the case studies of an organization. All the three critical values of the framework have been examined successfully with the organization's KPIs. As a result, the framework, when examined with the organization's case study gives a clear picture about what were the strong and the weak areas of the organization's KPIs regarding the effectiveness, the alignment, and the completeness and thus it shows in which areas the KPIs should be further improved by the organization. The evaluation results proved as an indication that the framework has a high potential to be used as an effective tool to help organizations detecting the missing KPIs.

4
Research Methods

In this chapter, we present the research methodology applied. In Section 4.1, we describe the objectives of the study and the underlying research questions. Section 4.2 elaborates on the case study approach that we have applied in our study. Section 4.3 introduces how we have selected the participants of the study and the domain exploration. In Section 4.4, we present our data collection strategy with the result of the interview transcripts. We also elaborate on how we have analyzed the collected data in Section 4.5. As a result, we have derived certain themes and then a qualitative model from it. Section 4.6 introduces the development of an alternative solution and its evaluation is reported under Section 4.7. The reporting of findings and possible threats to the validity of the research is presented in Section 4.8.

The research methodology, Figure 4.1, shows the overall picture of our research works. It explains various phases of our research in sequence-wise where each phase represents its own road map with a certain aim in it. The first phase of the research is *domain exploration and literature review*, in the case company, as to create a good understanding of the foundation of the research. Simultaneously, We have made a literature survey by reviewing the literature related to the domain, complex systems in particular. The summary of the findings of both these tasks helped us to create our first version of the interview guide tool. During exploring the domain, we explored the verification and validation (V&V) levels of the case company in the product development process. Since the overall research aimed at investigating of quality overview in different product levels of the development of self-driving cars, understanding each level in aspects to V&V activities, visualization of V&V, continuous integration & verification, aspects related to V&V's results in an integration level and aspect related to an end-to-end quality overview, was important.

The second phase was the *selection of participants* that comes after the domain exploration. This phase is as equally important as domain exploration because we are required to go through the organization stakeholders' structure and the corresponding verification level to select the appropriate participants for the study. After reviewing the organizational stakeholders' chart, including with support from industrial supervisors, we came to map the stakeholders that should have participated in the study. Finally, the list of stakeholders was summarized.

After completing the above two phases, the research methodology leads to the third

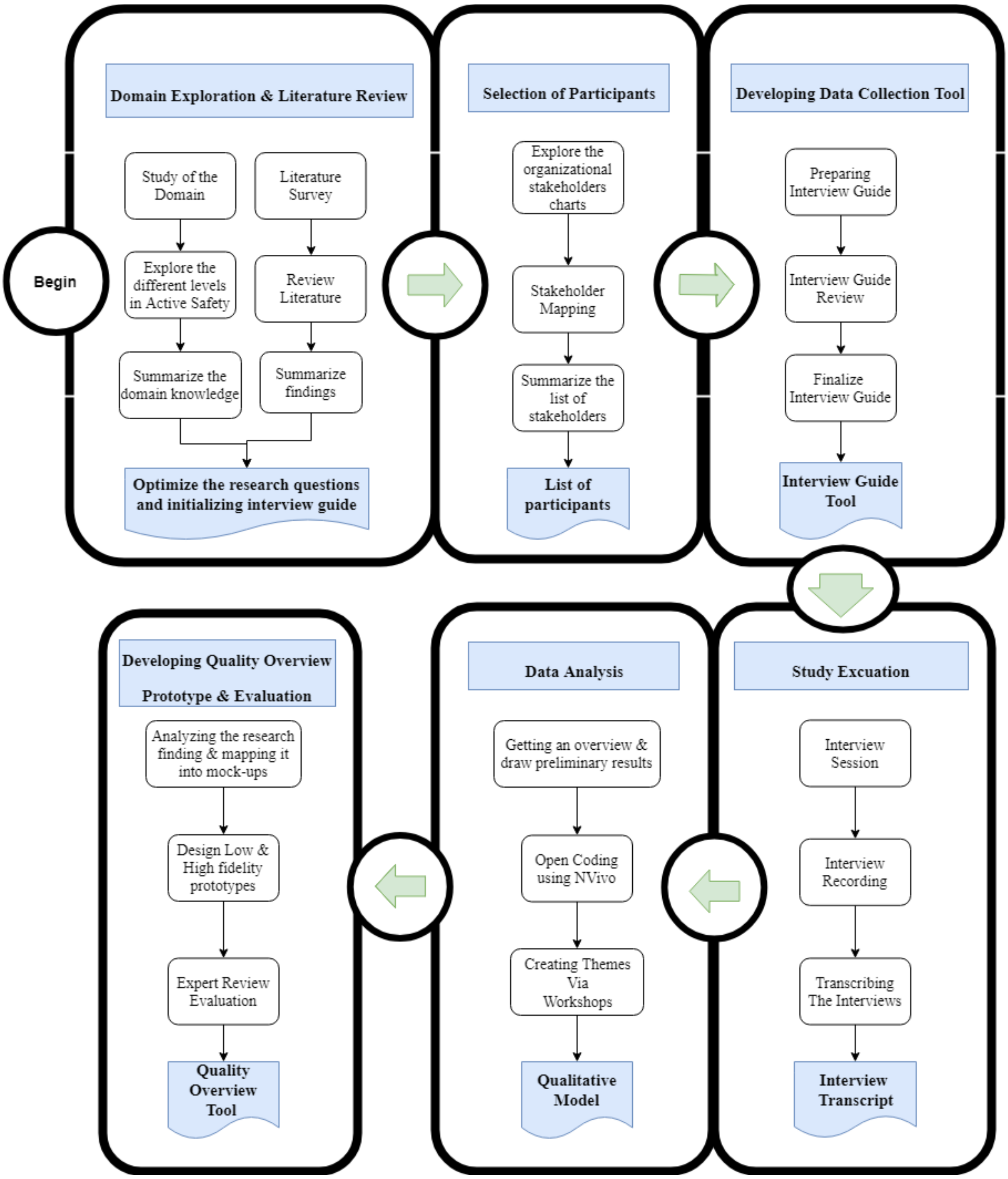


Figure 4.1: The steps of our research methodology

phase which is *developing data collection tool.* The term data collection tool refers to the interview guide which acts as a main source or tool to collect the data, as our research is a qualitative study using the semi-structured interviews. During this phase, we have done the initializing of the interview guide tool (via phase 1, Domain exploration and literature reviews), reviews (review of the interview guide by supervisors from the academia and industry), and produce the final interview guide as a result. The fourth phase of our research methodology is *study execution* where the data collection is conducted through interview sessions followed by the transcribing of the interview recordings.

The fifth phase is the *data analysis* where preliminary results are drawn as we walk through the interview transcriptions to get an overview. This was an important step toward developing the qualitative model and later an alternative solution tool. Therefore, we have drawn a preliminary research finding from the interview transcripts before doing the open coding of the transcripts. The preliminary research findings were used as feedback for the followup interview sessions. Then after all interviews being conducted, we went through open coding of all interview transcripts, all at once, using a tool called Nvivo[1]. As we got plenty of codes, from the 16 interview transcripts, in corresponding with the different aspects of the study, categorizing these codes into certain themes or categories were important. As a result, we created certain themes out of the codes during an industrial workshop where participants from industry and academia were invited. Then, a qualitative model, as shown in Figure 5.1, was mapped from the identified themes as per the research objectives.

The last phase was proposing an alternative *quality overview solution* where we analyzed the research findings and mapping it into mock-ups that could be used as a solution after the mock-ups went through low and high fidelity prototypes design. As per the validation of the alternative solution, we went through an expert review evaluating, using selected expert users from the case company, of the proposed tool.

4.1 Research Objectives

In this section, we will describe the research objectives of our study. The main goal of this study is to investigate the challenges and opportunities of current ways of delivering a product quality state overview, identify the main key performance indicators(KPIs), and stakeholders' quality overview visualization preferences. To do this, we have collected certain challenges, opportunities, and valuable KPIs and visualization choices from different stakeholders in different product integration levels/scopes at the complex domain through in-depth interviews. Based on this, a convenient quality overview, along with good KPIs and visualization preferences, is proposed to overcome the summarized challenges.

[1]https://www.qsrinternational.com/nvivo

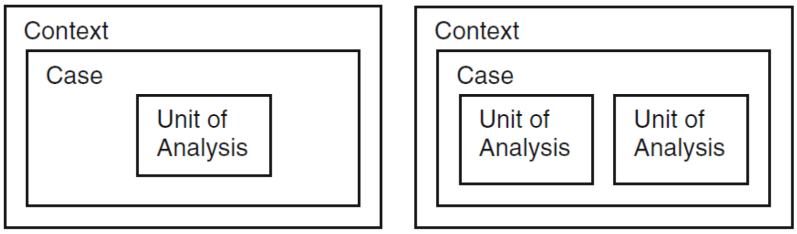


Figure 4.2: Holistic case study (left) and embedded case study (right)

4.2 Research Design: An Exploratory Case Study

In this section, we provide the rationale behind our choice of the research methodology, qualitative approach, and conducting a case study.

Conducting Case study

Runeson and Höst [13] argue that the general definitions of case study, as per the agreement of Robson [28], Yin [29], and Benbasat et al. [30], is an empirical method aimed at investigating a contemporary phenomenon in their natural context. Thus, considering our research objectives (See Section 4.1), we chose to conduct a case study.

As Runeson and Höst [13] argue, the same way as Yin [29] did, a 'case' is actually may, in general, be virtually anything which is a contemporary phenomenon in its real-life context. A case may be an individual, a group of people, a process, a product, a policy, a role in the organization, an event, a technology, etc [13]. Every 'case' has one or more units of analysis. Yin [29] distinguishes between holistic case studies, where the case is studied as a whole, and embedded case studies where multiple units of analysis are studied within a case, see Figure 4.2. In our study, as we have roughly four verification levels as a unit of analysis in the self-driving domain of the case company, we implemented an embedded case study with multiple units of analysis. In a case study [13], the case and the units of analysis should be selected intentionally. In contrast to other research methodologies, such as surveys and experiments, where subjects are sampled from a population to which the results are intended to be generalized [31, 28]. For that reason, we took the self-driving domain of the case company and its corresponding verification levels, during the development of self-driving cars, as a case and units of analysis respectively.

Case study research methods: exploratory, descriptive, explanatory and improving

According to Robson's [28] classifications, we have four types of research methodologies that serve different purposes.

1. Exploratory — finding out what is happening, seeking new insights and generating ideas and hypotheses for new research.

2. Descriptive — portraying a situation or phenomenon.
3. Explanatory — seeking an explanation of a situation or a problem, mostly but not necessary in the form of a causal relationship.
4. Improving — trying to improve a certain aspect of the studied phenomenon.

In this study, we did not intend to examine any theory. Instead, we aimed to understand the current ways of producing product quality information across various verification levels, while we focused on what challenges and opportunities exist in the current ways of doing. In addition, we aimed to identify the key performance indicators and stakeholders' product quality visualization preferences across the various different verification levels. Therefore, considering our research purpose and questions, we found the exploratory case study is the most suitable research methodology for this study.

Qualitative and Quantitative Approach

As per a definition provided by Creswell [5], we have two approaches for research methods to collect and analyze the data.

1. Qualitative approach — is an approach that involves emerging questions and procedures, data typically collected in the participant's setting, data analysis inductively building from particulars to general themes, and the researcher making interpretations of the meaning of the data.
2. Quantitative approach — is an approach for testing objective theories by examining the relationship among variables that can be analyzed using statistical procedures.

As we, in our case study, tried to understand the phenomena and identify useful distinctions that clarify our understandings using our research questions, specified in Section 4.1, which belongs to the category of exploratory questions [32], as a tool, We did not know all relevant variables of the phenomenon or any sort of hypothesis/theory prior to the study. For this purpose, we took a qualitative approach as a suitable research approach in our study.

To sum up, in this study, an exploratory case study is conducted, with a major focus on in-depth and iterative interviews. As case studies are flexible and may contain elements of other research methods [13], literature reviews are also done for the purpose of strengthening the interview contents (defining interview questions). Furthermore, an on-site case company exploration (observation) with a certain archival analysis is done as part of the data collection on the study.

The reason we needed to do literature reviews and domain exploration (archival analysis) is to ensure the triangulation [13], in a theory/viewpoint perspective.

In summary, the procedure, we went through to conduct the study is presented as follow:

- Stakeholder mapping (analysis of stakeholder - whom to interview)
- Planning interviews and defining interview questions.
- Holding interviews, collecting responses, and defining follow-up interviews (if necessary).
- Study the interview results.
- Open coding - data analysis
- Creating themes - data analysis
- Develop a qualitative model that summarizes the challenges and opportunities of the existing quality overview, KPI desired by stakeholders, and their quality overview choices.
- Develop proposals for how to design and implement the results of the interviews/research into a quality overview solution (prototype).

4.3 Domain Exploration and Selection of Participants

In this section, we describe how we have selected our participants of the study in our case study. As Creswell states, "the idea behind qualitative research is to purposefully select participants or sites [...] that will best help the researcher understand the problem and the research question" [5]. For this reason, we have considered the following assumptions to select appropriate interviewees for our study.

- Stakeholders from Unit levels: Software developers, Software deliveries acceptance test responsible, and etc.
- Stakeholders from Component level: ECU level testers, developers, test rigs' responsible, and project owners.
- Stakeholders from Domain levels: Domain level testers, developers, HIL rigs' responsible, and test strategist.
- Stakeholders from System levels: System-level testers, developers, box-car responsible, Complete HIL responsible, and test strategist.
- Stakeholders from Continuous Integration (CI): Both participants from a domain or local CI and VCC CI visualization (Volvo cars CI, company-wise CI).

The main reason why we have interviewed participants with different roles is as a way to triangulate the data, but also allows us to explore the domain from different viewpoints.

Table 4.1 gives an overview of the interviewees, their product integration levels/scope, and experience. In the last column, the roles of the interviewees are listed, together with the years of experience in the respective roles. The interviewees had worked in their current roles for at least 6 months.

No	Role	Level/scope	Experience
1	Software module tester	Unit	2.5 years
2	Acceptance Test Engineer	Component	2 years as a system tester
3	ECU Tester	Component	Around 2 years.
4	Project owner (PO)	Domain	1 year as a Product Owner.
5	Verification architect	Domain	3 years as a System Architect
6	HIL Rigs, Tester	Domain	2 years as a HIL developer
7	Domain Responsible	Domain	15 years in automotive industry.
8	System architect CI	Unit/Component	1.5 years as a CI Architect
9	System architect CI	Unit/Component	1.5 years as CI Architec
10	System test engineer	System/HIL	3 years as HIL engineer.
11	CI responsible	System	2 years as a CI Responsible
12	Boxcar test developer	System	2 years as a PO in Boxcar.
13	Test strategist	Domain	3 years in the domain.
14	Tester/test strategist	Component	8 years (automotive + Telecom.)
15	Test expert	System	17 years in software testing.
16	Test expert	All levels	10 years in verification Engineer.

Table 4.1: Survey of participating interviewees in the case study

4.4 Data Collection

This section presents how we collected the relevant data for the purpose of the study. According to the research plan, we have conducted semi-structured interviews [13], to collect qualitative data, as it would be open for improvisation and further exploration [13], that would also align with the planned iterative interviews. Thus, pre-prepared interview questions are being used while considering improvement of questions is applicable. Since a case study follows a flexible design strategy [13], the courses of the interviews meant to be varied to a certain extent across different interviewees.

In total, we have conducted 16 interviews, 15 individual interviews, and 1 group interview, with 19 individuals from the different scopes of the self-driving domain (unit, component, domain, and system product levels just to name it) in a total period of 8 weeks. We have invited around 21 participants through email and proceed with 19.

The interviews were individual interviews with only the researcher and the interviewee present. In one of the interviews, however, we have conducted a group interview together with four interviewees. The reason why we decided to do this was that they came from the same scope of working and had the same role with quite a similar level of experience. To mitigate the possibility of one interviewee dominating the interview, we followed posing of questions directly and addressing each of the interviewees. The environment among the interviewee was friendly and valuable, no one was afraid to state the truth.

Section 4.4.1 presents how we have prepared the interviews. Section 4.4.2 presents the techniques we used during the interviews to collect the relevant data. Finally, in Section 4.4.3, we describe what kind of tasks and steps that we have performed on the collected data, after the interviews, to proceed with data analysis.

4.4.1 Before the Interviews

After the tasks of domain exploration and literature reviews, we prepared the interview guide which is planned to be applied during the interviews, with semi-structured form. The interview guide consisted of both open-ended and specific questions. To ensure that as if we elicit the possible response that can cover all the research questions, we tried to annotate each question, in the interview guide, with the corresponding research questions of the study. Hence, before the interview sessions were being held, preparing a good interview guide tool that could answer the research questions was important. To strengthen the interview guide so that it can bring good research findings, we took the interview guide through the review of participants from both industry and academia. Therefore, the interview guides were both carefully reviewed by participants from the academia and case company, who have expertise in empirical methods and knowledge of a quality overview for verification activities respectively. The interview guide also goes through tests, several times before the interview sessions, to refine the interview questions.

The interview guide can be found in Appendix 1. It includes a general introduction of the researcher, the purpose of the study, the assurance of confidentiality, and gaining permission to record the interview session. The interview guide then continues with open-ended questions asking the interviewees to share their background, role, experience in the domain, or similar and across company and teams. Open-ended questions, such as *"Tell me something about your role in this company?"*, are used to make the interviewee interact nicely about their background and create a good connection between the researcher and interviewee. Hence, the open-ended questions had an important role in allowing new ideas to emerge. We also asked closed-ended questions, such as *"which scope/level are you working at?"* and *"in which integration level you are best at, other than your primary one?"*, for the purpose of getting quicker information about the interviewee and manage the interview accordingly.

In the interview guide, the interview questions are structured into multiple categories that apply a funnel model [13], starting with open-ended questions and getting more concrete afterward.

4.4.2 Interview sessions

During the interview session, we followed the interview guide tool and discussed almost all topics that appear in the guide. However, we tried to keep flexibility also with respect to stakeholders from different verification levels of the domain. A qualitative interview was conducted. It is considered as the most common and one of the most important data gathering tools in qualitative research[35]. The interview type was semi-structured, it contained flexibility and was improvised mainly depending on the interviewee's answers, and new questions were spontaneously constructed based on it if it contained interesting information.

In all interviews, we started by giving an introduction to the thesis, supported by the graphical representation of the research scope, verification levels in particular, which is gathered from the domain exploration in the case company. Hence, we borrowed an architectural view of the verification activities in the domain called Volare (See in appendix 5, Section D.1), as it gives a better understanding of the research scope and which aspect we are really focusing on. Moreover, we also, not often, try to present our preliminary findings when we feel some misunderstandings occur with our interviewees, regarding the research objectives. Hence, we tried to present our preliminary findings so that the interviewees could get a better understanding of the purpose of the study.

At the end of the interviews, we asked the participants for their ideal quality overview tools, stated as "what would your ideal quality overview look like?" and "How do you see quality overview working out 5 years from now?", on the purpose of collecting important input for the development of an alternative quality overview solution which was planned to be done after developing the qualitative model from the research findings.

4.4.3 After the Interviews

After all the interviews are conducted, we have transcribed the interview recording and made it available for data analysis. Then, we draw preliminary findings/theories which some of them we noted to be incorporated in the follow-up interviews, as we have already applied the flexible research approach [35]. The noted details during the transcriptions are applied during the data analysis session to support the result. Furthermore, after every interview, we requested the participants to give feedback upon the summaries of what is collected so far from them, which is also referred to as a member checking [12]. Thus, we have contacted back the interviewees and provide the summaries of our findings and accommodate the responses for the follow-up interviews. Hence, We have presented our preliminary findings in the follow-up sessions, with other participants, to receive feedback and explicitly engage our participants.

4.5 Data Analysis

In this section, we discussed the data analysis method that we implemented in our study. The collected data were analyzed using thematic analysis [20, 15, 13], as case study research is a flexible research method and corresponding qualitative data analysis methods are commonly used. The basic objective of the analysis is to derive conclusions from the data, keeping a clear chain of evidence [13]. The chain of evidence means that a reader should be able to follow the derivation of results and conclusions from the collected data[13].

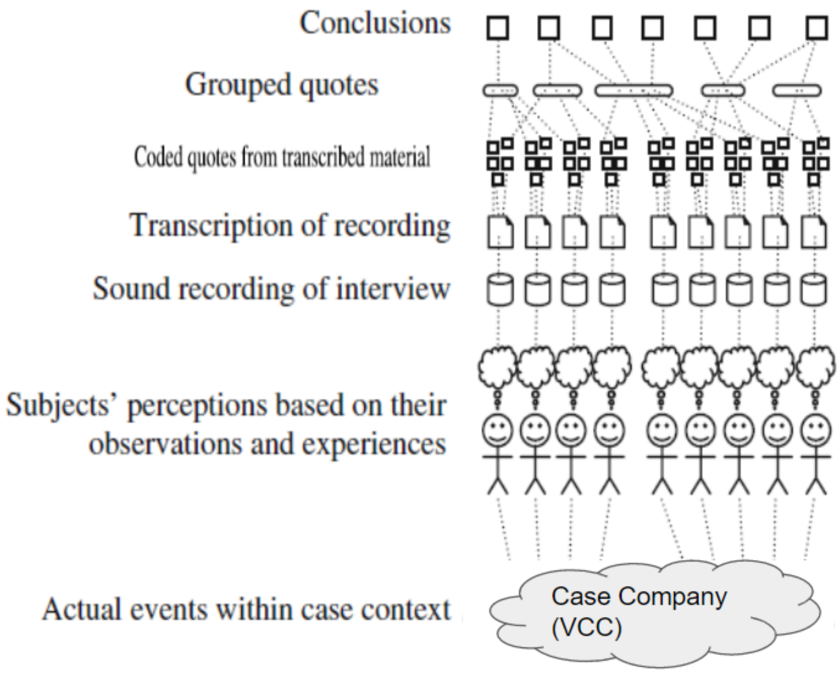


Figure 4.3: Data analysis approach adopted from a study [6].

There are two different parts of data analysis of qualitative data, hypothesis-generating techniques and hypothesis confirmation techniques [13], which can be used for exploratory and explanatory case studies, respectively. While hypothesis generation is intended to find hypotheses from the data, hypothesis confirmation is techniques that can be used to confirm that a hypothesis is really true, e.g. through analysis of more data. Therefore, the fact that we have tried to explore the domain to uncover the possible challenges, critical KPIs, and visualizations methods for the purpose of answering our research questions, we have basically conducted exploratory case studies with qualitative data analysis for hypothesis generation.

As a result, we have recorded the iterative in-depth interviews, transcribed them, and made them ready for the coding. We have applied a qualitative data analysis tool called NVivo 12 for open coding, creating themes and creating theories. As a data analysis approach, we have adopted a data analysis approach by [6] as shown in Figure 4.2, and tailored it to our needs as described in the follow-up sub-sections. As can be seen from the above figure, Figure 4.3, we collected data from different subjects of the case company. Then having the interview recording, we went through transcription and understanding of the results which is followed by coding of the transcriptions and grouped them into quotes or themes that could finally lead to independent results, which is called a qualitative model.

The remainder of this section presents our analysis steps: In Section 4.5.1, we describe how we got familiar with the collected data and eventually draw preliminary results. Section 4.5.2 presents how we continued the analysis using an open coding technique. Section 4.5.3 describes how we derived or created categories (themes) from the codes by the data analysis tool and the research findings in general.

4.5.1 Getting an Overview

After the transcription of the interviews is over, we tried to read the transcript several times carefully and collected the gathered notes and memos. We discussed some ambiguous phrases between the research team and took a common understanding. We took notes on the general ideas, meanings, and credibility of the interviews. This was to get familiar with the data and make sure we understood it completely.

Then we created preliminary findings by collecting the most important findings, just via a sort of agreement between the research team. As a result, this practice allowed us to get a deep understanding of the interviews and as an input for the follow-up interviews.

4.5.2 Open Coding

Before jumping into the process of coding data, [19] it is important to think about the research question and the big picture, which some may refer to as "storyline" or "meta-narrative." The story-line essentially belongs to the research objective which refers to rolling with the research questions, such as "What are the data telling me that will help me understand more about the research question?" [19]. Therefore, our research questions were our guiding tools while creating codes. Hence, following the transcription of recordings and a follow-up iterative walkthrough to develop a story-line or preliminary research findings, the raw data in the transcriptions are converted into the *"codes"* which refers to categorizing the text chunks from an interview transcription and labeling them with different terms [7], using a data analysis tool called Nvivo. Codes are usually used to retrieve and categorize data that are similar in meaning so the researcher can quickly find and cluster the segments that relate to one another [19].

Selecting a suitable coding tool was also a kind of important decision especially when the research team consists of two or more researchers. We have tried to use a couple of tools, just to name, NVivo, and Microsoft Excel. Despite some limitations on the merging of codes, we found out that NVivo provides a systematic platform for forming themes and giving codes than the latter. So, in order to minimize the effect of the merging problem in the tool, we decided to use the one common NVivo platform for coding of our data which is accompanied by an iterative discussion by the two researchers while coding.

For the coding process, we followed the guidelines in [19]. In the remainder of this section, we describe each of the steps in the coding procedure in further detail:

Reading through the data and creating a story-line/preliminary findings

The first step is to read through the data, several times, and develop story-lines or preliminary findings so that we remember the research questions while coding. Hence, we have read the transcripts and draw preliminary story-line which helped us decide what concepts and themes we want to communicate in our analysis. However, we didn't relate every code with the preliminary story-line.

Categorizing the data into codes

After creating the preliminary story-lines by reading through the transcripts, we categorized the data into certain themes using a predetermined coding [19], which is based on the interview guide or list of research questions and a previous coding dictionary from another researcher[22] or key concepts in a theoretical construct.

Using memos for clarification and interpretation.

In this step, we refined the codes as per our memos and informal notes. Therefore, after having coded the transcripts, we read through the codes and confirmed that there is no duplication of codes and identified if we need more categories to describe the codes in a better way before we create the themes. One of the more practical uses of memos is to record how you are developing the codes and making decisions about coding [24]. This enhances the audit trail to demonstrate to the reader how decisions were made, and conclusions were reached.

4.5.3 Creating Themes: Identifying research findings

This step deals with arriving at our research findings: Analyzing the codes and creating themes. After the open coding is carried out, we committed on organizing an industrial workshop on the purpose of categorizing the created codes into certain themes, such as challenges of current quality overviews, opportunities, areas to improve, key performance indicators, and visualization methods. Hence, during the industrial workshop, we have created themes and sub-themes in corresponding to the research objective. We did this using the following steps.

Analyzing Codes - Creating prior themes

Using our three main research questions as a basis, we identified prior themes that are of interest to our research. Those are challenges, opportunities, areas to improve, verification activities, existing quality overview, key performance indicators, and visualization preference. Then we create sub-themes and structured these codes as a tree, which can be seen in the figure below, Figure 4.4. Therefore, analyzing of codes, to identify interesting themes and sub-themes, is done before working on the themes.

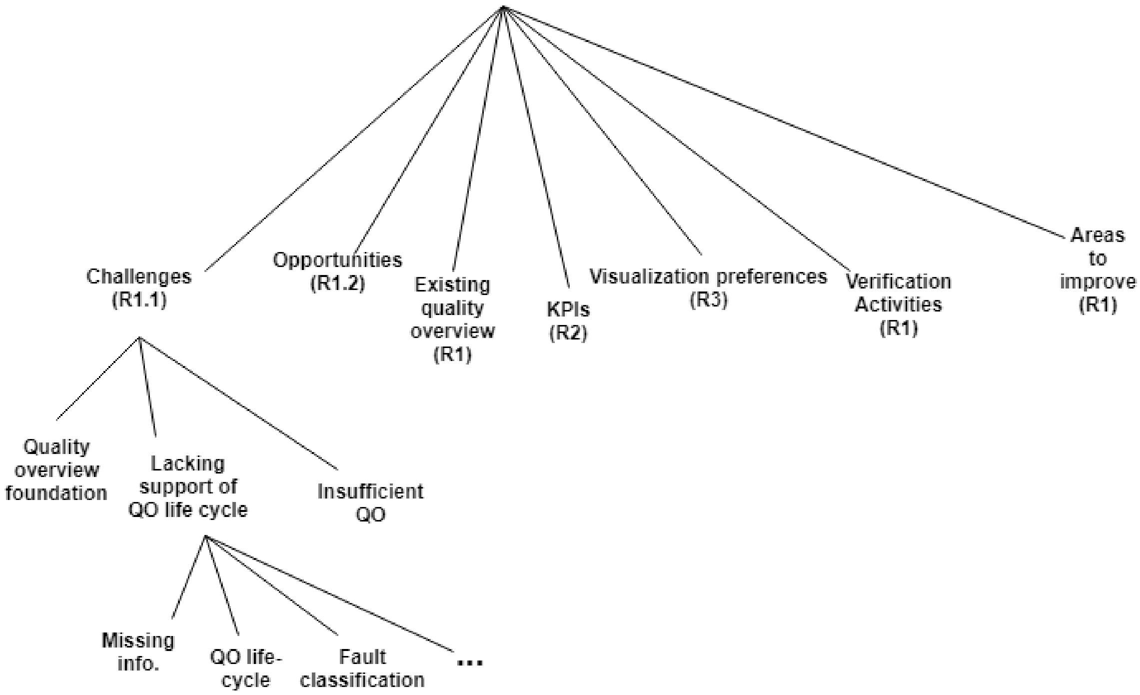


Figure 4.4: Excerpt of our a prior main themes.

On the highest level, the themes represent categories with regards to our main research questions. On the lower levels, we use themes to reflect the different concepts aimed to answer our research questions.

Creating a qualitative model

In order to create the final research findings, qualitative model, we create themes and categories from the analyzed codes through an industrial workshop, with participants from both academia and industry. The involvement of participants from both sectors allows us to make convenient conclusions on the research findings as we got different viewpoints [13] on the topic.

During the workshop, we printed out the codes along with brief descriptions on small sheets of paper, as it can be shown in Appendix 2, in accordance with the prior main themes. Then our research team presented the codes, descriptions, and the prior main themes to the participants. The workshop participants grouped the codes on different categories which is accompanied by a discussion of a possible connection of codes. Finally, the participants came up with interesting themes. This way, we synthesize the qualitative model from the themes built during the workshop.

4.6 Proposed Solution

In order to validate our research findings, we have developed an alternative proposed solution tool, using a prototype development tool called Figma [2], through the process of identifying the current quality overview challenges, opportunities and quality overview attributes (such as KPIs and visualization preferences) that remain around the verification and validation of the products produced by the case company. Hence, we developed a quality overview solution as an alternative solution for addressing the challenges discovered, KPIs, and visualization attributes which are broadly described in the following section of the report, Section 5.

While developing the alternative solution, we have gone through designing of low-fidelity and high-fidelity prototypes as we have to consider the design feedback between the iterations of designs. The term 'fidelity' describes how well it resembles the final product [26]. In the end, appropriate evaluation of the proposed solution is being done.

4.6.1 Low-fidelity prototype design

A prototype is a working model built to develop and test design ideas [25]. Prototypes that are closely similar to the final product are called high-fidelity prototype while those less similar are called low-fidelity. The first and most important role of low-fidelity prototypes is to check and test functionality rather than the visual appearance of the product [38]. A high-fidelity prototype is often made with the same methods as the final product and hence has the same interaction techniques and appearance as the final product but is more expensive and time-consuming to produce than a low-fidelity prototype [25].

We did a low-fidelity prototype design of the proposed solution using a paper design with a range of hand-drawn mock-ups, as it can be shown in Appendix 3, section C.1, which also went through a usability test by the research team. This design was helpful in enabling early visualization of alternative design solutions as it provokes innovation and improvements. After we were certain that we had carefully polished the design and accommodated the necessary functionality [38], we did the high-fidelity design. The high-fidelity design can be seen in Appendix 3, Section C.2.

4.6.2 High-fidelity prototype design

We built the high-fidelity prototype using a web-based tool called Figma. High-fidelity prototypes are a very close representation that appears and function as similar as possible to the actual product which is effective to demonstrate the actual implementation [26, 25, 38]. The design went through certain iterations so that we included all the necessary research finding that is summarized in the qualitative model. We took the design through functionality evaluation, also, to make sure it

[2] https://www.figma.com/

fulfills tasks related to a specific level overview, end-to-end, and stakeholder's perspective overview.

In conclusion, the final design consists of several functionalities, with more than 30 windows, that help the user on getting a quality update on all verification levels, end-to-end overview, project-wise overview access, testing environment status overview, and product-level overview.

4.7 Evaluation

To validate if the proposed solution is meeting the research findings, we took the solution prototype through an expert-review evaluation where expert users from the case company were as participants. The evaluation essentially consists of usability and functionality evaluations.

4.7.1 Expert evaluation

Before walking through the evaluation questionnaires, see in Appendix 3, Section D1, we made a demonstration of the proposed solution, to our participants, by presenting the three factors. Those are verification level-specific overview, end-to-end overview, and stakeholders' visualization preferences.

The expert-review evaluations were being conducted to measure the usability and functionality of the proposed solution prototype against a predefined scenario case tasks prepared by the research team. The participants were from the case company that has relevant experience in verification levels of the domain.

4.7.2 Expert evaluation tool - example task scenario

The research team prepared an expert evaluation tool to match the proposed solution prototype with the predefined example task scenarios so that to measure how much the proposed solution prototype meets the research findings. Using the expert evaluation tool, the participants ranked each question on a 5-point Likert scale, those are: Strongly agree, Agree, Neither agree nor disagree, Disagree, Strongly disagree.

4.8 Reporting on the Findings

We have created a qualitative model from the themes developed from the open coded transcriptions. The reporting of the findings of the study can be seen in Chapter 5. And In the end, we discussed the results of this study along with respect to literature and identified what new points popped up in the study which is described in Chapter 6.

5
Results

Based on the qualitative data analysis, we yielded 7 themes and multiple subcategories under them (See Figure 5.1) which we present in this chapter. We have illustrated our findings with quotations from the interview transcripts. The *Verification Activities* and *Existing Quality Overview* are the themes that serve as a foundation for our study where the challenges of current quality overview (Research Question 1) are investigated. These themes show how the current verification tasks, present in the case company, used to verify the product at different verification levels and its corresponding current quality overview tools, respectively, which is also elaborated further in the background study (Chapter 2) of this report. As can be seen in the figure, Figure 5.1, the *Verification Activities* theme connects with each theme as a source to answer every research question. Based on our exploration of verification activities and existing quality overview, as two of the themes in the study, we identified several challenges in the existing quality overview which is being represented as the theme *Challenge* and is described in Section 5.1.It gives answers to our first research question.

Along with analyzing the challenges of the existing quality overview, we have uncovered one more theme called *Opportunities of the existing quality overview* as the second part of our first research question. The reason why we explored the opportunities of existing quality overviews is to study what is the good thing on the current tools and subsequently accommodate it into the proposed solution. This theme is presented in Subsection 5.2.

To enrich the research question, Research Question 1, with valuable results, we have also created the *Areas to Improve* theme as a result of coding all interview transcripts in regard to the areas to improve, in the case of current quality overview tools, besides their opportunities and challenges. The theme is presented alongside with opportunities in Section 5.2. This theme provides useful insights and suggestions to all the research questions.

To achieve the enabling of a good quality overview (or to answer Research Question 2 and 3), we also explored two more themes which act as a quality overview enabler or attributes. i.e. *Key Performance Indicators (KPIs)* and *Visualization Preferences*. They are presented in Sections 5.3 and 5.4 respectively. It was noticed that technical infrastructure, work-flows, culture, and company context influence all the themes as well. Eventually, in Section 5.5, a proposed quality overview solution along with its evaluation results is presented.

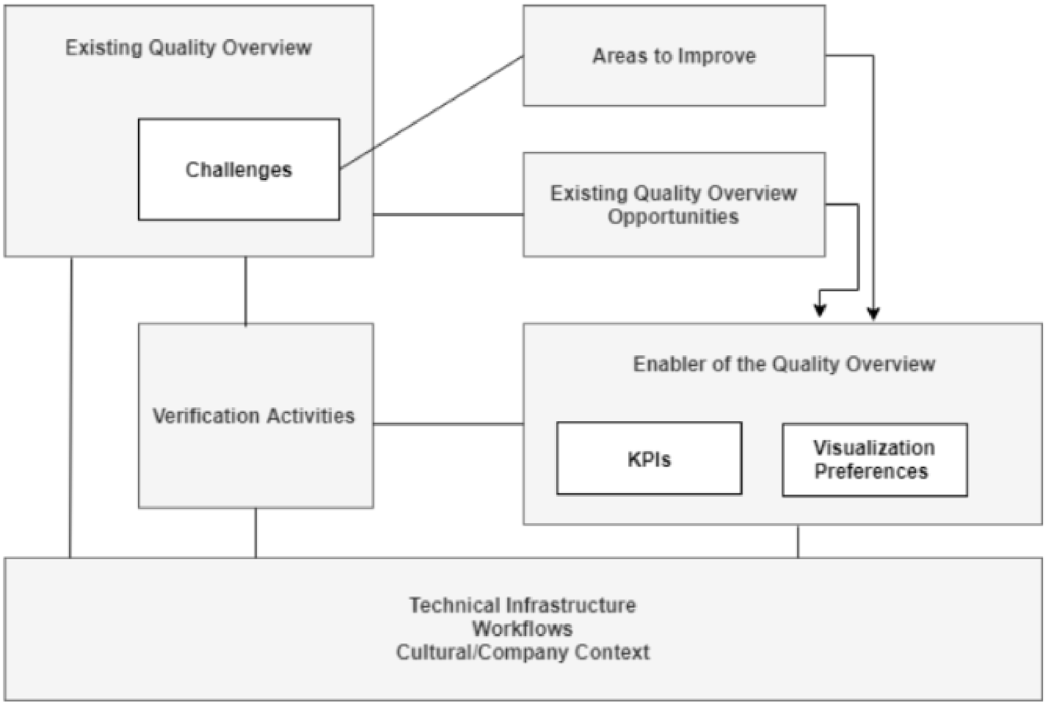


Figure 5.1: Our basic main themes of the research findings

5.1 Challenges of Current Quality Overview Approaches/Tools

This section introduces the challenge of current quality overview tools and ways of conveying product quality state information in the domain. Figure 5.2 and 5.3, shows an overview of the relevant aspects in the context of the challenge. In this section, we illustrate these aspects.

The first one is a quality overview foundation lacking challenges, through the development process and ways of working, which we describe in Section 5.1.1. In Section 5.1.2, we present challenges in regards to lacking support of quality overview life cycle. Section 5.1.3 presents challenges in regard to insufficient quality overview.

So, we have three major categories in challenges as can be shown in Figure 5.2.

- Process does not produce information about quality - Lacking quality information about the foundation of the quality overview life cycle through the development process and ways of working.
- Existing information not used - Lacking the support of quality overview life cycle.
- Insufficient quality overview - Lacking a good visualization.

As can be shown in the above figure (See Figure 5.2), each category has a chin effect on another. The lacking foundation of the quality overview affects the category in the middle (see Figure 5.2) by delivering insufficient quality state information,

Figure 5.2: The three major themes of the Challenges

which eventually causes lacking quality state support in the quality overview life cycle. The category in the middle has a chin effect on the *Insufficient Quality Overview* category because necessary information is not maintained or available as there is lacking support for the quality overview life cycle. Then, eventually, the insufficient quality information ultimately leads to insufficient quality overview and visualization. The different sub-themes of these themes will be presented as follow:

5.1.1 Lacking quality overview foundation

This theme uncovers the challenges regarding the lacking of quality overview foundations (Quality overview enabler) during the product development and its corresponding verification activities. Hence, this challenge primarily results in delivering of insufficient quality information for stakeholders in each product level or an integration level. Those challenge could caused due to the existing ways of working (WoW), poor presentation of verification result and development process.
As it can be shown in Figure 5.3, this theme has six sub-themes. Those are challenges related to Software module delivery issues, product testing level, testing environment status, test automation level, ways of working and others.

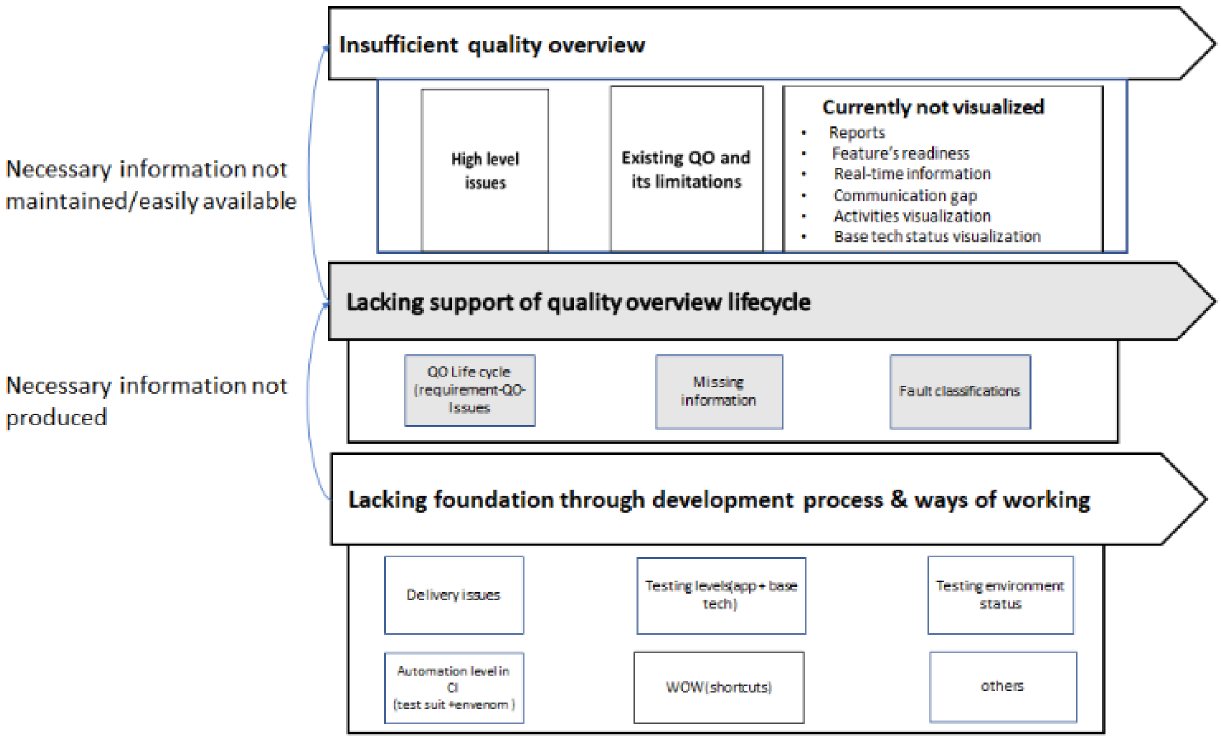


Figure 5.3: Challenges of current quality overview.

5.1.1.1 Delivery Issues

This category uncovers the challenges regarding software product deliveries from suppliers. Despite the deliveries arriving on-board, the less frequency of delivery and its delay hinder the current quality overview from achieving or producing a good quality overview. One of the consequences of regarding delivery issues challenges, in the case company, is elaborated in the following paragraph.

Freezing of requirements occurs due to high lead time between deliveries. Software module deliveries are often delivered every six weeks and in between. As a result, requirements get frozen. Thus, stakeholders are forced to fix the issues regarding failures in the software modules at a later stage. The following excerpt comes from the unit level stakeholders discussing the scenarios:

"*Suppose we get some software modules now delivered to us, and we get the other after six weeks. The lead time is often between six to eight weeks. While we receive the first delivery and wait for the next , the requirement might be frozen in the middle. This means if we find some bugs in the first delivery, we have to wait for the next delivery which might force us to wait several weeks to fix bugs we find today.*"

– Product Owner

In conclusion, in the case company, most deliveries occur every six weeks which leads to insufficiently tracking of the quality state trends.

5.1.1.2 Testing levels (base tech along with application testing)

This category uncovers the challenges regarding product testing levels and their corresponding testing strategy. Some product development levels appear to have problems in not incorporating all possible verification activities. For example, base-tech testing is missing in all product levels prior to System-level (or Boxcar testing). Base-tech testing is only done on system level via Boxcar testing. This might cause a delay in product development due to an accumulation of workload in a specific product level, system level. As a result, it is convenient to have both base tech and application requirement verification at all levels to optimize product development and time to release. Quality could also get improved by detecting issues earlier than before it reaches the system level. This argument could be supported by the following claim from a system architect on the domain.

“*What is happening now is that, unfortunately, we check the Base-tech requirements of every level in one level, system level. Thus, this becomes hectic to identify problems earlier.*” ’

– System Architect

5.1.1.3 Automation levels in CI (Test cases + Testing environment)

This category uncovers the challenges of current CI on automating testing activities. It also deals on automating testing environment settings, for verifying products to test automatically and get quick feedback. Hence, we have two sub-categories in this category. Those are the automation level of test cases in CI and the automation level of environment settings in CI.

Test cases automation into CI

This sub-category discovers the challenges toward the automation levels of test cases in CI flow.

To consolidate a good quality overview, automating test cases for both functional and non-functional requirements is important. However, in the case company, automation of both functional and non-functional requirements are missing.

Furthermore, in the existing CI flow, automation of application test cases along with base tech test cases is missing especially at component and domain levels.

Testing environment setting status

This sub-category discovers the challenges toward the automation levels of testing environment settings in CI flow.

Automating testing environment settings/rigs to CI flow encourage stakeholders, with products under test, to grow confidence in the testing environment settings. However, testing rigs are not automated into the existing CI flow, in the domain under study. This shows that automation in the CI flow has not been achieved. For that reason, test rig developers often find it difficult to gain the trust of their customers/testers on the quality status of the testing environment.

As suggested by one of our interviewee, quotes below, a good platform that could show the testing environment status, testing preconditions, status and confidence level for testers/developers is needed. The platform shall generate a real-time report which includes the real-time status of the testing rig (HIL), failure history, and testing preconditions. Moreover, test scheduling shall also be implemented on the platform.

"*It [the suggested platform] will be used as tools for test scheduling that shows availability of testing environment rigs and automatically inform testers the status, confidence level and prerequisites of the testing rigs.* "

– HIL Engineer/ Complete car

Furthermore, a platform with a feature that shows testing environment status where resource availability could be assured is desired by many test rigs/HIL developers. For example, in the current ways of working, HIL developers need to manually check for a software update. They need to ask the function developers or suppliers regarding the latest software update.One of the interviewees has stated this issue in his own words below:

"*It would be important if software delivery and update are automatically placed into CI flow or in to some common platform so that we[HIL developers] would be able to download and update easily.*"

– HIL Engineers/ Group Interview

5.1.1.4 Using shortcuts during verification (Skipping testing levels)

This category mainly refers to the challenges related to stakeholders' way of working during the verification activities. Some stakeholders intentionally or unintentionally skip certain verification activities or fail to send proper quality information and resources to stakeholders beneath or upper to their product level. The challenges in this sub-theme can be described below.

One of the challenges is some developers send test cases in an unorganized way, and some don't use existing tools to manage requirements. For example, some stakeholders don't want to use a tool called System Weaver, despite the fact all product requirements have already been written there. These problems often made HIL rigs' developers plan less efficiently.

Another issue is skipping testing environments. In the case company, according to data from the interviewee below, testers skip HIL rigs and show a tendency to directly test in cars. Aside from the fact stated by our interviewees, testing in HILs provides a cheaper, faster, and convenient way to test than in actual cars, skipping any intended verification activities contribute negatively to the quality overview tool needed.

"*The one thing which I wanted to discuss before coming to this meeting is a lot of the function testers, at Active Safety [domain], are interested in testing directly in the car [skipping the HILs]. It should go through HIL and then to the car. So, this one thing which we care about because HIL is the cheapest and the fastest so that we can reduce so many expenses than we have Cars and have to drive millions of kilometers for autonomous testing.*"

– HIL Engineers/Group Interview

5.1.1.5 Others

This category mainly combines the other uncovered challenges in existing product development. These can be related to tool licensing, some intercultural issues, or administrative issues. For example, some current quality overview tools (such as System Weaver) are not open for all stakeholders.

Another challenge is that even if there are limited tools to show quality overview, they seem to not be user friendly. Stakeholders often get difficulties in eliciting the quality state of the products under test. The below excerpt shows the challenges more broadly.

"*First, you need to have access to System Weaver. Not everyone has that because we have a limitation in the number of licenses. This is the first step basically which prevents us. I mean most people like project leader, manager, or other roles, do not have access to the System Weaver. Due to this, they cannot look into the testing results. The second part is how do you find the quality results in the System Weaver, it is not easy to find, you have to go to the right place in the tree structure which is really hard because you don't know where to go.*"

– Release Train Engineer

5.1.2 Lacking support of Quality Overview Life Cycle

This section introduces the theme under challenges which are related to the current quality overview life cycle, during project/product development, such as handling requirements, dealing with faults, and working with information flow. This causes to insufficiently maintain the existing quality information in the product verification life cycle and prevent the ease availability of quality information for a stakeholder to pursue them.

As it can be shown in Figure 5.3, this section has three sub-themes. Those are current quality overview life cycle, missing information, and fault classification.

5.1.2.1 Quality Overview Life Cycle - Requirements and its challenges

This theme represents the challenge in the context of the current quality overview life cycle. It mainly focuses on challenges regarding test suit (test cases) stability, mapping of test suit (test cases) to abstraction level information, its measurement, and trace-ability of test cases failures issues. The following are some results from the study related to this sub-theme.

To verify each requirement along with the requirements' coverage, there should be at least one test case synced with each requirement. In the current scenario, however, the number of test cases is not that relevant as compared to the coverage of the requirements. Hence, the mapping of test cases into requirements is lacking.

The stability of the testing suite (confidence level percentage) for different releases in the same delivery or baselines is lacking. Any existing quality overview tool should have shown the difference in test results for different releases in the same delivery. So that the stakeholders would be able to check the stability or confidence level of the test suite, and identify whether it needs an improvement plan in the test suit or testing strategy as a result. Hence, such a tool is lacking or not available as per the need.

During the interview session, a concern regarding the trace-ability for the issue is addressed. For example, when the base tech tester reports an issue at JIRA, it is difficult to track the issue. In order to overcome this problem, an issue tracking platform for stakeholders is required. The existing tools are not sufficient for such challenges, as can be seen partly mentioned by the following interviewees.

"*If I see something is failing, I need to see if somebody is taking care of it or it is just a new error.*"

– Product Owner/Boxcar

Another challenge in regards to requirements is legal requirements. Legal requirements' coverage is also missing along with the function's requirement coverage. Legal requirements are a kind of standard for vehicle implementation.

5.1.2.2 Missing information - while presenting quality information

This theme highlights challenges in an area where sufficient information flow is missing. This happens when the desired quality information flow and support from various stakeholders regarding test results are missing. For example, when the test results are not communicated sufficiently to stakeholders at the integration level. And, if stakeholders at an integration level do not provide good support to the other levels which eventually leads to missing information. The following are the key points from the study related to the missing information flow theme.

When we have interviewed the stakeholders from the unit level, we have found out that the generated acceptance test report does not give sufficient information for the next integration levels other than unit or component as it contains low-level information, mostly related with technical details which could not be understood by the stakeholders at the upper level. Therefore, the generated test report from unit and component levels via acceptance testing does not seem to be useful to the next integration levels in the existing way of presenting test results.

Good quality information flow is lacking also due to the lack of proper tools for online collaboration and common platforms to find software deliveries and others to work with. For example, the HIL rigs' responsible are not getting the required things for the test from other stakeholders in time, and thus this leads to delay for the HIL rigs test results and for the HIL testing. The lack of a common platform for collaboration, to achieve good quality information flow, can be described briefly by the HIL developer below.

"*If you need one software from an ECU, then you manually contact the developers for downloading and if we want test case for a function, we need to do the same.*"

– HIL Developer

5.1.2.3 Fault classification

This theme uncovers the challenges for the systematic approach (or using a platform) that can show or visualize the failure analyzing reasons along with failure categorization, showing that the failure happens due to the product, integration of products, test cases, or the environment settings. This challenge can be described concretely by one of the Software developers, in the domain, as follow:

"*A vehicle is a complete component, so if you test function A, you are not certain that the other team came and did something with the function B, and that might impact your test and if you could be able to track what is happening and where then the vehicle management is always needed with very strict control of the version.*"

– Software Developer/Domain level

This shows that a proper failure categorization is missing, which could manage the functions' failure analysis results along with its visualization. The consequence is that if something fails, it is hard to determine the cause behind the failure. The same concern applies to the HIL (testing rig), HIL developers are responsible to immediately determine whether the failure is a software error or a HIL error. In a complete car, if test cases are failing, the reason behind the failure of test cases should be analyzed whether the test cases are failed due to the product or the test cases itself.

5.1.3 Insufficient Quality Overview

This section introduces the themes under challenges related to the required good quality overview and visualization across the various product development. Thus, this section discusses the limitations of the current quality overview, and how to visualize the current quality state.

As can be shown in Figure 5.3, this section has three sub-themes.

5.1.3.1 Existing quality overview/tools and its limitations

This sub-theme uncovers the challenges of the existing quality overview for the purpose of providing the complete picture of the quality state during product development. By this sub-theme, the study identified the limitations of the existing quality overview along with consequent improvement suggestions.

The first challenge is that the current quality overview does not incorporate manual verification activities, such as legal requirements, as conveying the quality state of products. As a result, few verification activities get visualized on the existing tools. This causes insufficient conveying of quality information for stakeholders irrespective of their level. This argument can be supported by the excerpt for one of the developers.

"*If manual testing and driving testing results will be included in the overview then I will be so confident when I see that all those have passed.*"

– CI Developer

Another challenge is some teams are lacking real-time quality information as they utilize Agile events as the only source for providing quality overview information, and these events might not occur as needed. In this way, stakeholders might not be able to deliver real-time quality information. Scarcity of the real-time quality information increases the rate of fault slip through and thus it becomes difficult to make a quick decision and to track fault slip through at an appropriate level.

System Weaver as one of the quality overview tools, for instance, has all requirements of the products. However, the tool is lacking in providing sufficient information about some requirements and its coverage, which becomes inconvenient for some stakeholders.

Another tool is CI flow dashboard which acts as one of the ways to see the quality status where stakeholders with access can view the quality status per project and at different high-quality levels. However, the challenge is that the view is limited to quality information in a project wise way.

5.1.3.2 High-level quality overview is lacking - quality information in an abstraction level

This theme shows the absence of a high-level view of the quality state of products along with proper visualization. It introduces the challenges regarding a common portal that could provide a better understanding of the quality state for all stakeholders, which are supposed to lie at a different level.

While the study shows some visualization along with detailed technical information for product level to certain product levels, presenting end-to-end quality information at an integration level is lacking to a larger extent. This hinders some stakeholders in different integration levels to understand the quality information from the other. Thus, stakeholders in the integration level are unable to determine the status of the products. Due to this reason, stakeholders often manually need to ask the other stakeholders about the status which consumes time and ultimately leads to slow progress in decision making. The following are some results from the study related to this theme.

When an issue occurs in a particular product level, a stakeholder in another integration level might not understand the issue as everyone uses their own terms. Hence, having one common visualization could allow stakeholders to easily identify the issues and activities at other integration levels. This way, a common quality overview tool could provide a common language (picture) for everyone on an integration level, and this is currently missing in the domain under study.

The other challenge is, as a common portal for the quality overview information for every stakeholder is missing, quality information for products in different integration levels are scattered in different tools such as System Weaver, Jira, and etc. As all stakeholders do not have access to various tools in the case company, it is inconvenient for many stakeholders to follow product quality information at an integration level and end-to-end. Therefore, a well organized common portal for a quality overview in each scope and end-to-end is in demand.

5.1.3.3 Currently not visualized - Tools exist but not visualized for better reach

This theme uncovers the challenges regarding the current way of delivering quality state of a product with a good quality overview and visualization. Here, we aim to identify the challenges that hinder the current quality overview to visualize its quality and activities.

This theme has six sub-themes that cover different ways of providing product quality state status and activities. Those are challenges regarding visualizing various reports, feature readiness, communication gap, base tech status, real-time information, and visualizing testing activities.

Test Report

This category mainly uncovers the challenges regarding visualizing existing reports of products' verification results and their trends. For example, visualizing ECU failure reports across projects, integration scopes/levels in projects, and across testing environments is lacking. Overcoming this challenge would help in figuring out which ECU causes the main issue in the car. This argument can be strengthened by the following claim from a product owner in the system product level.

"*If one ECU is failing in the first project and is OK in the next, then again fail in the next project. This is hard to determine. A tool that can do something like this would help to see the history of each ECU failing across projects.*"

– Product Owner/Boxcar

Moreover, ECU software downloading status reports across integration levels is missing. There should be ECU software downloading failure status at each level so that it becomes easy for the stakeholders to identify the ECU failure status. Hence, visualizing these test reports graphically or any other sort would be important as it has explained by one of the interviewees in the study.

"*ECU name, say X, is OK in the component level, but the downloading failed at a complete level. So, there should be a graph, in the report, that shows ECU vs pass/fail status.*"

– Product Owner

The other challenge is regarding testing environment settings like HIL rigs. A proper report of prior health checks of HIL rigs is missing. This hinders developers/testers to build confidence in the testing environments as if the failure happened because of the testing environments or their products. This can be supported by the following response from one of our interviewees.

"*When you go to the HIL rigs. Many stakeholders utilize the rigs to run test cases at different times. So, we need information regarding the previous failure report of the rig before we do anything. If we found out, from the rig, that a particular software already had a problem. Then, we will put that into consideration. So, we rely on the reports but missing.*"

– Developer/HIL

Feature or Product readiness visualization

This category uncovers the challenges in providing a quality state overview at an abstraction level. Thus, stakeholders could have a 'bird's eye' view of the quality of products along with the product-specific level of quality information. For example, Stakeholders might be interested in which feature is ready to release rather than how many test cases get executed or how many requirements get fulfilled. Moreover, a platform is in demand for showing feature stability across different integration levels.

(Visualizing) Real-time quality information is lacking (E.g Agile)

This category uncovers the challenges of the current way of delivering quality state of products which possibly cause a delay in delivering and visualizing real-time information with valuable performance indicators.

A proper platform that delivers real-time quality information is lacking. One of the existing quality overview tools is an Agile event. And these events seem not to provide real-time information as it happened occasionally. This causes a delay in decision making and fault slip through issues.

Communication gap - (Visualizing) Communication between stakeholders/product levels toward verification is lacking

This category mainly belongs to the challenges regarding communication among various stakeholders, in an integration product level, and the corresponding verification activities to maintain a good quality overview between stakeholders. The following are the key findings from the study regarding the communication gap between stakeholders.

There is a need for establishing a kind of online platform which shows the test results with a feature that could help stakeholders in communicating online. Hence, a platform that creates good communication and collaboration among stakeholders is in demand. For example, HIL developers are interested in getting feedback from testers who utilized the HILs as it can be shown in the excerpt below. So, visualizing such collaboration and communication is needed for better decision making.

"*Developers come and conduct the test on our HILs.They take the logs and then they do not communicate back with us [the HIL rigs responsible] after testing their product. Or there are no ways that we know our HILs are properly working, other than the health check we do regularly.*"

– Developer/HIL

Verification/Testing activities visualization is lacking

This category uncovers the challenges regarding visualizing testing activities with respect to the level of detail. The current quality overview lacks complete visualization of testing activities. Only a few testing activities are visualized. The following are key points in the study regarding this category.

For example, a tool entitled VCC CI visualization shows the status of the product for all levels of stakeholders but lacks complete visualization of the testing activities in the products at an integration level. A good visualization that visualizes all the testing activities, for the purpose of supporting stakeholders to quickly find in which arrears they should focus on, is lacking.

Base-tech requirement status visualization in all integration level

This category uncovers the challenges regarding visualizing Base-tech testing activities at each product level. Visualizing of Base-tech requirements and its corresponding results is missing in all integration levels.

5.2 Opportunities and Areas to Improve

5.2.1 Opportunities

This section introduces some opportunities for the current quality overview tools and ways of conveying product quality state information in the domain. In this section, we have explored different opportunities that can be categorized into three themes. Those are activity enabling overview, release enabling overview, and real-time information overview.

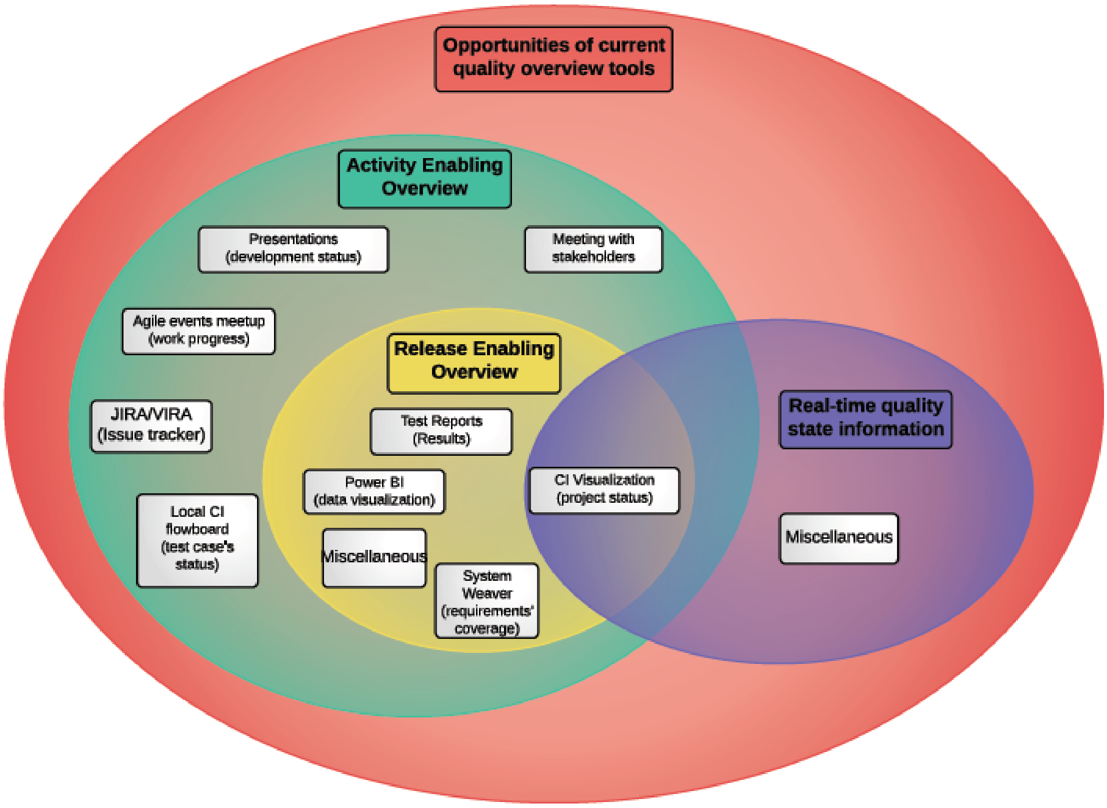


Figure 5.4: Summary of opportunities of existing quality overview tools explored during the study.

Activity enabling overview

This theme uncovers some opportunities for the current quality overview that might enable certain verification activities. These existing quality overviews provide a good overview that allows the stakeholders to proceed with their upcoming activities. For example, some HIL rigs show readiness information via an excel sheet report, so that the developers would be able to grow confidence and trust in the testing environments. Some examples of the tools are described below.

The first one is VIRA-JIRA is a kind of tool that acts as an issue tracker. Every issue is assigned for a stakeholder to fix. Thus this helps stakeholders in viewing the results. This facility is available for those customers who test their products in HIL and deliver the reports through this tool in the case company. This tool is also connected to the other tool in the case company, System Weaver. Using this tool, the tester can see the test cases and its corresponding results. If some fault occurs then it is recorded as an issue in VIRA-JIRA.

Others are the test reports via excel sheets for HIL which describes the readiness of the test environment settings (HIL) along with details. These reports enable the upcoming verification activities as the result of HIL is to be determined.

Release Enabling Overview

This theme uncovers some opportunities for the current quality overview that might enable releases. Some current quality overview tools give certain information that could trigger releases by showing quality status and information regarding failure trends. For example, System Weaver shows the release history of the pass and fail criteria. Thus, this helps in predicting and gaining insights regarding the releases. The following are some of the tools or activities that help in enabling releases.

System Weaver, a tool for keeping track of the requirements' coverage shows and generates a release history of the pass and fail criteria and observes it from release to release. Besides, it can build charts and show pass or fail criteria with automation levels. With the help of such features, stakeholders can view the release history more easily. This argument can be supported by the following:

"*Like in the third release we have 80% fail and 20% pass and in the fourth release we have like 40% fail and 20% pass so that you can see how software evolves during the time and if it has positive trends or negative trends.*"

– System Tester/ADAS

Another important tool that acts as a release enabler, by showing the quality information, is Power BI. It is a Microsoft tool that collects data from all levels and visualizes it on a web browser. Using this tool, one can create their dashboard as per their wishes like a customize dashboard, thus helps in enabling releases.

"*Visualizing the CI will continue but this power BI also contains the other visualization where you can import CI results here and from other systems as well, you can import and get a good overview.*"

– Release Train Engineer

We used 'Miscellaneous' labels for a collection of tools that stakeholders barely use in this category. There are main tools which we have mentioned above under this category. However, for some other tools which are barely used by stakeholders, we put them under the "Miscellaneous" label.

Real-time Information

This theme uncovers some opportunities for the current quality overview that could enable the real-time quality for products in some product levels. Some current quality overview provides certain real-time information about product verification, process, and product results. For example, VCC CI visualization provides the quality status of the project for every stakeholder.

VCC CI visualization provides the quality status of the project for every stakeholder. Stakeholders may view the latest status of the project online and which baseline they can use. It also gives information about the quality of the test case level. By viewing the latest status of the project one can easily identify the baseline to use within a second.

We used 'Miscellaneous' labels for a collection of tools that stakeholders barely use in this category. There is main tool, VCC CI visualization, which we have mentioned above under this category. However, for some other tools which are barely used by stakeholders, we put them under the "Miscellaneous" label.

5.2.2 Areas to improve

This section introduces the areas to improve in the current quality overview tools and ways of conveying product quality state information in the domain. In this section, we have explored mainly three bigger sub-themes for areas to improve in the current quality overview foundation, life cycle, and visualization.

Areas to improve in quality overview foundation

This theme introduces the areas for improvement where the current quality overview foundation shows limitations. These can be areas related to the existing verification activities, product development, process, or way of working. Base tech and HIL are the two main areas mainly focused on the improvement.

Base-tech testing exists only in the boxcar level while it is convenient to exist in all product levels (unit, component, and domain) along with application testing. Therefore, Base-tech test cases in each integration level with different requirements are needed to improve the verification activities on the boxcar testing.

Regarding the HIL rigs, there is a need for the platform which shows the status of the HIL rig before testing. The platform should generate a real-time report which incorporates the status of the HIL, failure history of the HILs, and testing preconditions. Therefore, a systematic platform is needed in which developers (testers) send testing packages to HIL rigs so that HIL rigs could provide the best testing environment. Also in regards to WoW (way of working), few stakeholders skip the HIL test and go directly to the actual car testing whereas HIL rigs have many advantages as compared to the actual car testing.

When it comes to testing environment status, a tool for test scheduling is recommended. With the use of such a tool, testers could automatically send tests with certainly required capabilities and thus it would get routed to the correct rig or put in the CI queue. Therefore, proposing a tool that is used for test scheduling in all rigs would be beneficial for both the testers and rigs' developers.

Areas to improve in current quality overview life cycle

In this section, we have explored the areas to improve regarding a good quality overview life cycle during product development and verification. Under this theme, we have a couple of sub-themes that uncovers the areas to improve on the current quality overview life cycle. Those are quality overview life cycle and fault classification.

- **Quality overview life cycle**

  This theme deals with improvements regarding the current quality overview life cycle. This mainly focuses on how to improve the quality overview life cycle in activities like keeping track of failure, requirements, features, functions, and test cases. For example, during software delivery from third party suppliers (unit level), the software is tested against the case company requirements. But there are also unwritten requirements to be considered by the acceptance tester. Therefore, the current quality overview should hold all requirements in different formats such as written, model, and unwritten requirements. There should be both functional and non-functional requirements to be mapped with test cases for the quality overview life cycle improvement.

- **Fault Classification**

  This theme uncovers the areas to improve in relation to analyzing failures during verification activities. When failures appear during verification activities either in products or while testing on environment setting, stakeholders often experience challenges regarding failure analysis. For example, during failure's root cause analysis, stakeholders often seem to struggle whether the failure is due to the product, test cases, or environment settings. If a test case result is failing then in order to determine the reason for the failure of test cases whether it is faults in the test cases or the product, connecting test case results (failure) to failure analysis results would help in quickly determining the failure analysis. In addition to the above area to improve, the quality overview should have a platform especially for HIL rigs that shows automatic failure categorization i.e. product, test suit, or environment settings.

Areas to improve in the existing quality overview visualization

In this section, we have explored the areas to improve regarding creating a good quality overview and mitigate the existing limitations toward visualization. Under this theme, we have a couple of sub-themes that uncovers the areas to improve on currently not visualized quality overview tools and product's high-level quality issues.

- **Currently not visualized**

  This theme mainly focuses on improvements regarding visualizing an existing way of delivering a product's quality state to provide stakeholders with a better-quality overview and visualization. It shows a proposal for improvements in relation to quality overview, status, information flow, real-time information, and feature readiness.

  When it comes to reporting, ECU wise report in addition to a project-wise in a complete car (boxcar) is needed, which is illustrated by the following words of one of our interviewees.

  "*If one ECU is failing in the first project and OK in the next, and then again fail in the next project. We can see the history of each ECU failing across projects.*"

  – Product Owner

  Software downloading is one of the major struggles that stakeholders at the boxcar level face. So, they desired to have ECU software downloading status reports at each level in order to easily analyze the situation if a software downloading test has failed.

  Another concern regarding the complete status of a project at each level should exist. For example, base tech testing and application testing should exist at each level along with its report. One of the interviewees has expressed concern.

  "*If someone wants to validate a project (for example 519A) then they might not be knowing the full picture of what the status of the project in their level is. Because when it comes to the unit, component, and domain levels, they can only view the application testing results, not base tech testing until the boxcar testing.*"

  – Product Owner/Boxcar

  Thus having the base-tech and the application test reports at each level would help in validating the full project in an easier and appropriate way and view its full status.

  The high-level view is of great importance when it comes to the quality overview of the product, especially for upper-level stakeholders. There is a need for the higher-level abstraction so that stakeholders can view the high level of a project like (release - feature -a requirement - test case) other than only test case level. The connection of test results to some higher-level abstraction would help to understand the status of the product in order to proceed forward and make decisions.

Good information flow about the product status across each level is necessary to have. In order to have it, there should be a convenient platform in the form of reports that could facilitate information (testing report) flow which should be maintained across testing environments.

All the testing activities should be covered in the quality overview. The quality overview should incorporate manual testing activities as well. In the current scenario, only automated testing activities are in consideration. For example, one of the manual activities is the function rating requirements, which should also be incorporated in order to have a clear picture of the product.

Apart from the above need in the case company. The stability of the testing suites if shown would help in organizing the test more effectively. With the aim of determining the stability of the test suite, there should be a feature readiness where the stability of testing suites can be judged easily for the different releases in the same delivery or baselines. One of the interviewees has expressed his concern regarding it.

"*If you get the same kind of delivery but new versions and you run the test, it would be very nice to see what is the difference between these releases.*"

– CI Architect

- **High-level Issues**

  This theme uncovers the areas to improve regarding high-level issues in the quality overview domain. Hence, this theme introduces the areas of improvement when it comes to the delivering of the higher-level quality state status.

  The quality overview should provide the stakeholders with an efficient way to understand the current state of the product in order to help them to understand where to invest their effort. So, there should be a scale in the quality overview of the problems which could show where an effort is needed.

  Having a good and portable visualization that delivers quality information regarding test cases' results trends rolling on a TV screen or phone would be convenient for the stakeholders.

5.3 Key Performance Indicators (KPIs)

This section introduces the identified needs for key performance indicators across the verification activities at an integration level and end-to-end. Figure 5.5 shows an overview of the relevant categories of KPIs categorized as fault analysis, requirement/feature, base-tech testing, automation and CI, test suit quality, product quality, and test environment quality KPIs, which are formed due to its various subcategories or causes which altogether feed to the high-level KPIs, representing the summary of all the categories in an abstract level. Further, we describe each category of the KPIs and its subcategories in brief.

The study's result for the key performance indicators is presented in a cause and effect analysis approach using Fishbone diagram. Using this approach, we tried to show how a high-level KPIs would be determined through the in-depth analysis of identified KPIs, in an integration level, and its effect in across product level via Fishbone diagram, as this diagram is appropriate for graphical representation of exploring and categorizing, clearly and simply, the potential root causes of the various events or problems [39].

5.3.1 Fault analysis KPIs

In this section, we refer to KPIs related to fault analysis at any product level during product development. Fault analysis related KPIs is important and desired by almost all the interviewee from all levels.Based on the data analysis, we have categorized this theme as "Fault Analysis" because it contains the KPIs related to fault analysis, i.e. resolution time, fault status (open, closed reopen, or resolved), faults appeared aftermarket and fault slip through across product levels.

1. **Turnaround time and resolution time**

   This KPI shows the resolution time for faults by stating an estimated time to resolve a particular issue and this KPI is mentioned by three of the interviewees The Turnaround time KPI is measured by the time taken for the completion of a particular verification activity. One of the interviewees has mentioned regarding the turnaround time KPI for verification activities –

   "*SWD perform various activities like software downloading, diagnostic testing etc. There should be a time duration shown for such activities and the reason behind is stakeholders are mainly interested in results out of verification activities, they usually do not care about the turnaround time for the verification activities until it becomes a big issue which could lead to slow progress.*"

   – CI Architect

2. **Test failure issue status**

   Along with the KPI for showing the resolution time for faults, it is important to view the status of the fault, which currently exists, in the case company, in the existing quality overview through VIRA-JIRA. This KPI is mentioned by the 3 of the interviewees. This KPI shows how many test failure issues are in a state of analysis, verification state, closed (resolved) or reopened, etc. According to the product owner at the boxcar level, it would be good to track the failure issue. This contributes to ease proceeding with the decision making according to the requirement.

3. **Number of Fault Slip through**

   This KPI shows the number of the fault slips across product levels. According to one of the release train engineers, it would be good if one could know which test cases did not pass in the previous product level. Hence, stakeholders could determine what problems are there in the previous level and be able to solve that in the lower integration level itself. It is always easy and effective to solve the problem at a lower integration level than finding its causes in the higher

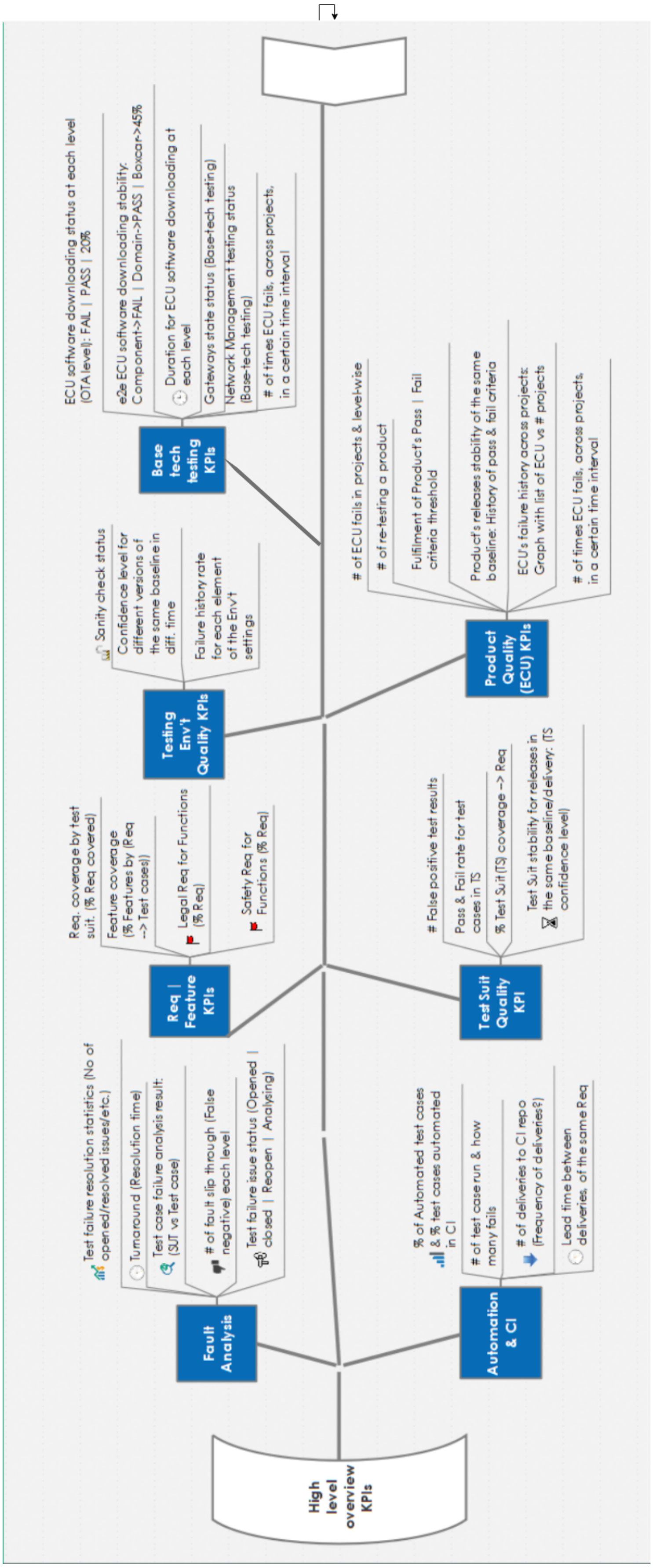


Figure 5.5: High level Overview KPIs

levels. This reflects the importance of a fault slip through as a KPI.This KPI is mentioned by 2 of the interviewees

4. **Number of opened issues**

   A number of open issues for vehicle projects under development and aftermarket is considered as one of the quality overviews where the solution department in the case company tries to make the conclusion of quality using this KPI and this KPI is mentioned They investigate a number of open issues for the project out in the market and the projects under development. This KPI would help in checking the stability of the test suit.

5.3.2 Requirement/feature KPIs

This sub-theme discovers the KPIs related to test suite coverage towards the requirement, features, and functions. It indicates if the functions (domain), features, and requirements are sufficiently covered by the test suit. The following are the KPIs under this sub-theme.

1. **Requirement coverage via test suite**

   This KPI shows how many requirements are covered by the test cases in the test suite. From the research findings, we conclude that requirement coverage is considered as one of the important KPI in the process of a quality overview, which leads towards the betterment of feature readiness. This argument can be supported by a developer from the domain as follow:

   "*We need to see not only how many software modules delivered, we want to see also to what extent the software modules covered the test suite the suppliers utilized.*"

   – ASDM Developer

   As a result, the coverage of test cases in a test suite by requirements which is followed by requirements' coverage to fulfill feature readiness ultimately leads to the betterment of the quality of products. So, this KPI supports stakeholders on tracking the requirements' coverage.

2. **Feature readiness via test suite**

   A KPI that shows if the test cases (in a test suite) are as many as it covers the feature's requirement. This KPI would help in answering this question – Do all the available test cases fully test my feature of a particular product?

Hence, this KPI shows if the features are covered well enough by the available test cases, in order to determine the amount of testing done on the feature and its quality.

3. **Functional (Domain) requirement fulfillment**

   This KPI shows to what extent a particular function or domain has assured requirement coverage. Hence, stakeholders will be able to make sure that all the functions according to the requirement are working properly or not. This argument can be supported by the following excerpt by a system architect from a domain level.

   "*Suppose we have a function [Active safety function - auto braking system] at hand and that function should break when a bike is coming to the street. Then, we have to test the car for auto braking to make sure the requirement is fulfilled or not. And, there should be a KPI that shows such thing.*"

   – System Architect

4. **Functions' domain legal requirements**

   A function needs to fulfill the authorities' requirements or legal requirements along with system or functions' requirements. The authorities have a kind of standard for vehicle implementation. Therefore this KPI is needed for the fulfillment of requirements from a legal perspective.

5.3.3 Level of automation and CI - Automation in CI

As CI flow used to see the test cases' results, it enables for quick feedback and presents a quality overview at a specific level. However, as part of our study, we have also tried to gather important information regarding KPIs for such overviews in CI flow and its corresponding automation level. Hence, this theme discovers the KPIs related to a level of test cases or test environment settings and automation in CI flow. The Following are some KPIs that belong to this theme.

1. **Test case automation level**

   From the data analysis, we have observed that most stakeholders desire to have all test cases automated to CI flow. Due to this reason, stakeholders need some KPIs that could show them the automation level of the test case into CI. Hence, stakeholders will be able to get a clear picture of how many test cases are automated into CI and how many are left, as part of the improvement plan.

2. **Number of test cases running**

   A KPI that shows the number of running test cases. Along with the percentage of automated test cases, the KPI could show the number of test cases passed or failed which could support stakeholders to drill down to the root cause of the failing test cases.

3. **Deliveries' status**

   A KPI that shows the delivery details which includes the number of software modules delivered into CI repositories. This can be illustrated by the following excerpts from a developer and solution train Engineer in the domain, respectively.

   "*It would be good to have a KPI that shows the number of software modules delivered into our CI repositories. So, we can follow the status and see which ones get delivered today.*"

   – Software Developer

   This KPI is mostly useful for the software test expert who needs to keep an eye on the delivery of the software and maturity level of the CI.

   "*Suppose we are working on a RADAR sensor and we depend on software modules delivery from suppliers. We need KPIs that show how often we get software modules like if we get it every six weeks, every week, every day, or every hour. How often they [suppliers] manage to deliver the software to us. This is interesting. We are also interested in the number of deliveries, quality of software, how many test cases are run and the number of faults found.*"

   – Solution Train Engineer

4. **Lead time between software deliveries**

   A KPI that shows the lead time between deliveries, and it is applicable in all integration levels of the product development. Stakeholders relate this KPI with requirements frozen which can be supported with the following excerpt from a product owner in the domain.

   "*Suppose we get the software today and after the six weeks, we get the other software. If the lead time has conspired, we might get our requirements frozen. For that reason, we need a KPI that track down the lead time of deliveries.*"

– Product Owner/ASDM

5.3.4 Test suit (test cases) quality KPIs

This sub-theme discovers the KPIs related to the testing suite, test cases, and the corresponding test results. The KPIs check the test suite's stability, a number of fault positive/negative results, and failure rate. In this section, we will describe the relationship between the test suit and the test cases and how it could help in determining the stability of test suits.

In all product levels, test cases are the first initial step to test for which the stability of the test suit is needed. Following are some of the important KPIs related to this sub-theme:

1. **Test case failure rate and coverage**

   This KPI shows the failure rate of test cases and coverage at all product levels. As can be seen in the following excerpt from a solution architect in the domain, coverage, and failure rate act as an important indicator of the fulfillment of requirements or features.

   "*One KPI which I would really like to see, and is missing, is the requirements or test coverage. I need to see the test coverage and failure rate analysis because if you have the test coverage of 1% and a pass rate of 100%, it is not the same as you have the coverage of 99% and the pass rate is 50%. So, I need a KPI that explicitly differentiate both*"

2. **Stability of testing suite**

   Along with the coverage criteria, the stability of the test suit needs to be measured. A KPI which could show the confidence level of the test suit for measuring the test case results is needed. This way, stakeholders will be able to determine the stability of the test suites across product verification.

3. **Number of false positives for function (domain) testing results**

   A KPI that shows a number of test cases failure while the function is right. This causes more effort into the testing strategy by showing misleading performance reports of the function.

5.3.5 Base tech quality KPIs

This sub-theme uncovers the KPIs related to base-tech testing in Boxcar and other integration levels. The following are some of the important KPIs related to this sub-theme.

1. **ECU's software downloading**

   This KPI shows the confidence level of software downloading at a complete level in the form of a graph.

   - **OTA level of ECU's software downloading**

     ECU's software downloading as one of the major tasks on every product level demands to have several KPIs which are necessary for the quality overview. One such KPI is - "ECU's software downloading status – OTA level status". This KPI should show software downloading state in each level with a status of fail, pass, or in the form of X% where X is any number between 1-99. For example, ASDM software downloading in the component is passed and downloaded only 60% on a complete level.

   - **ECU software downloading confidence level in complete/system level**

     This issue regarding software downloading confidence level is expressed by one of the product owners at boxcar level is as follow:

     " *... in a particular report, an ECU "X" is OK in the component level but it is failing in the complete/system level. The graph also shows that the X-axis is always green in the component level and then it goes off at the system level. Such visualization is needed but we currently do not have in the existing CI visualization.*"

     – Product owner/Boxcar

     In conclusion, there is a need for KPI, in the form of a graph, that shows a confidence level of ECUs especially across product levels.

   - **Time duration for software downloading in each level.**

     This KPI shows the time duration for the software downloading process of ECUs. This supports stakeholders in keeping track of the time duration for software downloading for the purpose of maintaining good progress and speed.

2. **Gateway's state for base-tech testing**

   This KPI shows if the message among buses is successfully established. The product owner at the boxcar level stated that the gateway is used to check if messages are delivered successfully from one bus to another. When the messaging starts, stakeholders can trace for the purpose of checking if it is following the correct channel, timing, and destination.

3. **Network management testing**

   This KPI shows network performance among the ECUs which would affect the major activities of the car and would also help in showing the power consumption. This argument can be broadly described by the following excerpt from a product owner of a boxcar.

   " *... For example, if you start the car then how much time it takes for the buses or nodes to wake up. When the car is switched off, how much time it takes for the buses to go to sleep mode. If it does not go to sleep mode, does it consume power?*"

   – Product Owner/Boxcar

5.3.6 Test environment quality KPIs (HIL quality)

This sub-theme discovers the KPIs related to testing environment settings and status.

1. **HIL self-check result status KPI**

   HIL developers do sanity checks every night to see whether all the basic things are in place. So, the KPI which shows the status of the testing environment before activating it for developers would help in increasing the speed of the testing process.

2. **Confidence level of HIL rigs due to failure categorization**

   This KPI shows the confidence level of HIL rigs. The confidence level of the HIL rigs shows how trustworthy a particular HIL is for testing the same software at different times or different versions of the same delivery. This argument can be supported by the following excerpt from the HIL product owner.

   "*It will boost my confidence if I know that my HIL is trustworthy, supportive, and helping testers to find bugs before it goes to the car and even before reaching the customer. Besides, I want my HIL to show the same results across different delivery of the same products*"

   – Product Owner/HIL

3. **HIL failure history for each product level**

   This KPI shows the failure history of testing environment settings across product levels. Showing the failure history for each of the elements tested in HIL would help in analyzing the status of the HIL environment more effectively.

5.3.7 Product or ECU quality KPIs

This sub-theme discovers the KPIs related to the product quality and its corresponding product verification results. The product can be a software module, ECUs, and functions (domain). This theme would help in effectively providing product verification status. The following are some of the findings regarding this sub-theme, which various stakeholders have desired.

1. **Number of ECUs fail at each verification level**

   An ECU as a product needs to have certain indicators to determine its performance. It is important and useful to know the number of ECU's that have failed. A KPI that could show the number of ECUs failed at each level and a whole project is needed, therefore.

2. **Number of verification of products at all levels**

   A KPI that shows how many times a particular product gets verified due to a failure that happened because of the product itself, its integration with a problematic product, or testing environment settings. This would help in determining the exact reason behind the failure.

3. **Fulfillment of a pass and fail criteria for software and ECU against the test cases**

   The fulfillment of pass and fail criteria means if the KPI threshold is a ratio of 90 to 10 (90:10), it should show the rate of 90% test cases passed and 10% fail. If this threshold is OK for the product or it has passed the fulfillment within this threshold to move into the next level, then it should move in, else it will be tested again.

4. **Product release stability of the same baseline**- history of pass fail criteria

   A KPI that shows the stability of a product at each level which consists of different releases of the same product or baseline and variation of failures could be compared across previous releases to predict the future release and trends. The stability of the product shows the consistency of the release of the same product or baseline.

5. **ECU failure history across projects**

   A KPI that shows the number of projects an ECU shows a failure status via a graph with a list of ECUs VS projects. This KPI would help in determining which ECU causes the main issues in the car, as it can be shown in the following expert from the product owner in the domain.

   "*We wanted to have some kind of report which shows ECU wise, not projectwise. For example, if ASDM is failing in a project identified by 519A, but then the same ASDM goes OK in another project. This shows we need a KPI that shows how many times the same product[ASDM] fails over the year.*"

   – Product Owner/Boxcar

6. **Total number of times ECU fails across certain projects in a certain time interval**

   A KPI that shows the behavior of an ECU across different projects in a particular time frame. For example, in the year 2019, how many times ASDM failed.

5.3.8 High-level overview KPIs

This sub-theme uncovers the KPIs related to a high-level overview of the projects at an abstraction level.

1. **Vehicle Project status KPIs**- Done, not done or delayed (time plan)

   A KPI which could show the status of the vehicle under the category – "Done, not done or delayed" along with its assignee. This KPI would basically help the system project leader (SPL) who is responsible for the vehicle project that must be deployed with all the systems. They also report the specific stages of that vehicle.

2. **Solution level product status KPIs** – Done, Not Done or delayed

   A KPI which could show the status of the product development under the category – "Done, not done or delayed" on the solution level along with its assignee. This KPI would basically help the product manager responsible for reporting on their product development and general status like what is done, what is delayed, and where is the big problem.

3. **Project-wise KPIs**

Number of failing tests, number of problematic features and problem categorization
A KPI that would show the following results project wise:

- **Number of failing tests**
  Keeping track of failing tests if it is increasing or otherwise.

- **Number of problematic feature areas**
  In a higher-level overview, a KPI is needed to show problematic features due to failing test cases

- **Problem categorization** (product, testing environment)
  If test cases are failing, it could be because of a problem in the product or in the testing environment, so that one would know where to put the effort.

4. **Project status test results** (base tech and application) for all levels.

   A KPI that shows how a particular project behaves at each product level by fulfilling the testing of base tech and application. The results can be shown as a green or red color.

5. **Aftermarket customer satisfaction KPIs**.

   A KPI that shows the customer response regarding the aftermarket issues.

5.4 Visualization preferences

This section describes the third research question - in what ways stakeholders prefer a quality overview to be presented? The figure below, Figure 5.6, is a mind map diagram of the visualization preferences discovered in the study. The visualization preferences have certain main themes in various ways or features desired by the stakeholders for the quality overview purpose.

The motivation behind using the mind map diagram to visualize the discovered visualization preferences is because it is considered as a great tool for inscribing a piece of gathered information on a page while showing the relationship between the concepts involved [33]. A mind map allows a user to record a great deal of information in the form of linked ideas and images [33].

The first sub-theme is *Technical features/ products*, described in Section 5.3.1, which stakeholders desire to have in the quality overview dashboard. In Section 5.3.2, the *Release quality overview* is described, which would help in enabling the activities for the product's release. Section 5.3.3 describes the sub-theme *Activity enabling visualization* which shows certain visualization preferences that could help in enabling the activities based on the status of the products or verification tasks. In

Section 5.3.4, the *Adaption per stakeholder* sub-theme is described as among the various visualization ways that are desired by the stakeholders. Adapting visualization to stakeholders' own way, a snippet from the main visualization dashboard which is open for all stakeholders is described in particular. *Bird's view of all verification activities and results* is also another sub-theme described in Section 5.3.5. This sub-theme supports stakeholders on viewing the verification activities and the corresponding verification results from a high-level view perspective.

5.4.1 Technical Features

This sub-theme describes the various technical features in the form of a quality overview dashboard in which stakeholders desire to have quality state information of their products. Such visualization preferences help in providing the technical details of products across certain product levels and verification activities.

Stakeholders desire to have an online visualization for a product feature's test results which is accessible to everyone. Abstraction level for such product feature's test results would help stakeholders, from various integration levels, to understand the quality status. Online visualization of the feature's test results, in an abstraction level, with an option of drilling down to the root cause analysis, could help stakeholders in digging deep down into the details and see what has caused the failure, as it can be seen from the following snippet from one of our interviewee's argument.

"*Pass and fail for the feature's test results are the two main factors that a stakeholder may want, to dig down into the details in order to find its causes and its criticality. And, its effect on a project level during the release of a product should be analyzed too.*"

– Developer

Some stakeholders expressed the need for a graphical presentation to show product failure which could also show failure trends for the purpose of hunting failure details on that particular aspect. This graph visualization would help stakeholders in finding the root causes of failures easily.

Another approach is a technical feature that could show the test cases result in a color-coding. Thus failing test cases can be represented by red color and green for the test cases made it to pass. Moreover, along with the color-coding for good and bad quality, failure analysis visualization of the bad quality would support stakeholders to easily get the root cause for the bad quality.

In conclusion, all of the above mentioned technical features that stakeholders desire to have, ultimately help in tracking the quality of the product and the corresponding testing environment settings for the purpose of providing quick and relevant feedback for every stakeholder.

5.4.2 Release quality overview

This sub-theme shows a visualization preference that could act as a release enabler. The quality overview visualization preferences under this theme could help in visualization of releases across different verification levels, release test result details for every level, tracking of release test failure issues, and test suite's result difference between releases of the same baseline.

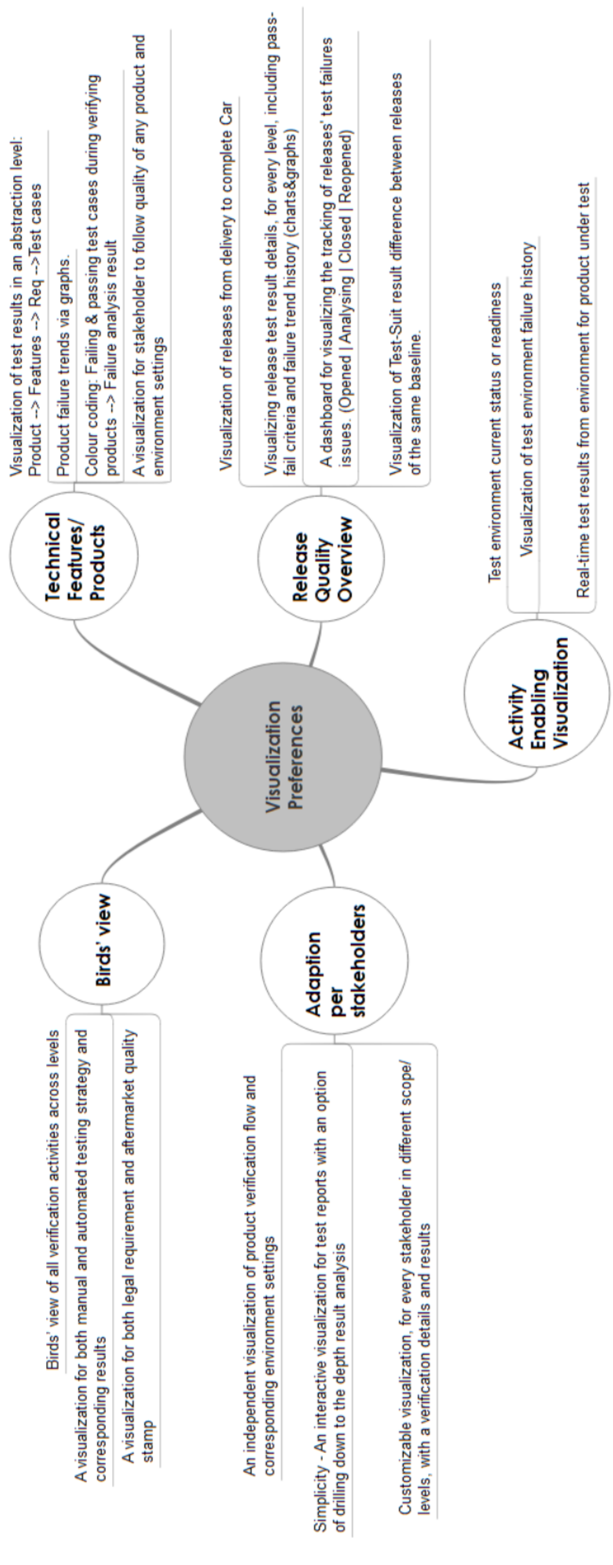


Figure 5.6: The mapping of KPIs explored during the study

One such visualization is showing releases from delivery to complete/system car. Such visualization would help stakeholders to track the release from any verification level. Moreover, a visualization dashboard that shows the summary of the test results that incorporate pass/fail criteria and release history using charts and graphs with a drilling down option is needed. Such a visualization dashboard would be appropriate for all product levels as these stakeholders would be able to manage the pass/fail criteria and release history in charts and graphs.

As a conclusion, a dashboard visualization for tracking the release's test failure issues stating the number of issues in analyzing state, verification state and closed, or reopened are desired by many stakeholders across product integration levels.

5.4.3 Activity enabling information

The visualization preferences under this sub-theme are used as activity enablers. In this section, we will discuss the visualization preferences for testing environments settings, HIL rigs, in particular, desired by stakeholders, that could also trigger developers to conduct their testing on the testing environment settings in harmony. Hence, the stakeholders need a dashboard that could show the test environment settings' especially HIL rigs status prior to the verification process, that shows the HILs' readiness.

Another approach is the testing environment settings should provide real-time test results for the product under test. Furthermore, a visualization that could show the HIL failure history would help in analyzing the failure reasons in a faster way.

5.4.4 Adaption per stakeholders

This sub-theme discovers visualization preferences that provide stakeholders to adopt their way of visualizing quality information about the status of the product flow and corresponding environment settings.
One approach is stakeholders seek for an independent visualization of product flow and corresponding environment setting to be maintained. This argument can be supported by the following excerpt by a HIL Engineer.

"*There should be two different views for quality state visualization, one is the official software view and the other is the rig view.*"

– HIL Engineer/Complete level

According to the above quote, a visualization dashboard that could show a detailed view of the software and the rigs separately. Having such a facility would be convenient for the stakeholders in a scenario where a software developer is not interested in seeing the actual details of the product but rather interested in the performance of the software. The same goes for the rig developers.

Another approach is bringing simplicity and interaction in the visualization dashboard by introducing an interactive visualization for test reports with an option of drilling down into the depth. Thus, a customizable visualization dashboard for every stakeholder in different scopes/levels of a project containing its verification results and details is required. Hence, stakeholders will be able to view the quality overview in their domain. This argument can be supported by the following excerpt by a Solution Train Engineer.

"*It would be very good if they have one common platform for all for the quality overview and stakeholders from different holders can customize from that one common visualization dashboard to their level where they can see the quality in their domain.*"

– Solution Train Engineer

5.4.5 Bird's view of all verification activities and results

This sub-theme discovers visualization preferences that provide stakeholders to view all verification activities along with their results from a bird's view perspective. Using this visualization, stakeholders would be able to view the manual, automated testing, and legal rating verification activities and results quickly.

One common quality overview entry point where a delivery can be traced into a car and both automated and manual stages should be visible and information and results from these stages should be reachable. Hence, visualization for manual, automated, legal requirements, and aftermarket quality stamps from a bird's view perspective are needed.

5.5 Proposed solution and its evaluation result

For the purpose of validating the research finding, we have developed an alternative proposed solution and subsequently evaluated it using an expert evaluation technique. The following subsections discuss the evaluation process and the results.

5.5.1 'Example Scenarios' Tasks

While the expert evaluation was conducted using the expert evaluation questionnaire (See Appendix 1, Section A.2) as an expert evaluation tool, we have applied an example scenario tasks for the purpose of eliciting if the proposed solution meets the research objectives, findings, and the needs. Hence, using this one approach of expert evaluation techniques, we tried to confirm, with the participants from the case company, that the proposed alternative solution fulfills the requirements of an estimated alternative quality overview solution.

In an example scenario task, scenarios are the main entity to deal with. Scenarios are narratives, usually in a written form, that describe how users use the conceived product to perform a certain task [26]. They highlight the goals of people that will use the product, what people will do with the product, what works and what does not work, and how people interpret what happens when they use the product [26]. Hence, to see how our participants conceived the proposed solution, we have derived three major example scenarios and certain tasks beneath them to provide an optimal representation of the usage.

The major categories of our example scenario tasks are end-to-end overview (along with KPIs), product level-specific overview (along with KPIs), and visualization preferences. The following is a detailed list of tasks in each category.

1. **End-to-end quality overview (Along with KPIs)**

    - Select project and see ECU status (and KPIs) and SDWL. [Proper ECU failure report for each project is lacking?]
    - Select a particular ECU and see its status (and KPIs) across projects and levels. [ECU's failure history across projects?]
    - Select testing environment and observe its status (and KPIs), preconditions and failure history?

2. **Verification level-specific overview (Along with KPIs)**

    - Select verification level and observe the status (and KPIs) and product quality? [Verification level-specific product quality overview?]
    - Stakeholders able to see a complete quality overview of product quality, i.e. if both manual and automated quality results are incorporated?
    - Stakeholders will be able to see if they meet base-tech requirements from their own verification level perspective.

3. **Variety of visualization preferences**

    - Quality trends via graphs, pie-chart and etc.? (Baseline version stability, ECU failure history, SWDL, etc.?)
    - Customized visualization of projects, products (For example, ECUs), and testing environments (testing HILs) that aren't available in their own verification level.
    - Visualization or product's quality overview in a hierarchical manner, from abstraction to low-level information.
    - Visualization of fault classification (with an option of drilling down to the depth?).

No	Questions	User Responses
1	It gives quality state of product in an efficient way?	Agree/Strongly agree
2	It seems realistic?	Agree
3	It should be implemented soon at the case company?	Agree/Strongly agree
4	It helps for stakeholders on their decision-making activities?	Strongly agree
5	It gives end-to-end quality overview?	Strongly agree
6	It contributes to informing product failures trends on the purpose of an improvement plan?	Agree/Strongly agree
7	It creates a good quality overview of products for different stakeholders' perspective at Active safety?	Agree/Strongly agree
8	The findings from the study are incorporated enough in the solution?	Agree
9	It is an important enabler for large scale agile way of working? [Frequent delivery with quick feedback and data-driven decision making]	Agree/Strongly agree
10	It allows stakeholders to track the product quality through the different integration levels?	Strongly agree
11	It meets the expectations about the kind of the quality overview you wish to see as per the completion of this study?	Strongly agree
12	Does it meet the scenarios in section 5.5.1?	Agree/Strongly agree

Table 5.1: Expert evaluation questionnaire response summary.

5.5.2 Expert Evaluation Result

Two experts participated in the expert-review evaluations. These experts had previously been interviewed during the main data collection of the study and participated in the theme formulation workshop. Hence, this contributes to the expected evaluation validity threats due to the lack of awareness from the participants on the research objectives.

As it can be shown in the table below, Table 5.1, the answers for each question from the expert-review evaluations, where the choice was Strongly Disagree, Disagree, Neither agree nor disagree, Agree and Strongly Agree, shows a combination of the latest two choices (agree and strongly agree). This implies that the proposed solution can be considered as one of a candidate alternative solution that at least answers the above questions appeared in the evaluation session. The answers for the questions in the expert evaluation questionnaire can also be summarized as follow, Figure 5.7, which indicates almost half of the questions got Agree/Strongly agree.

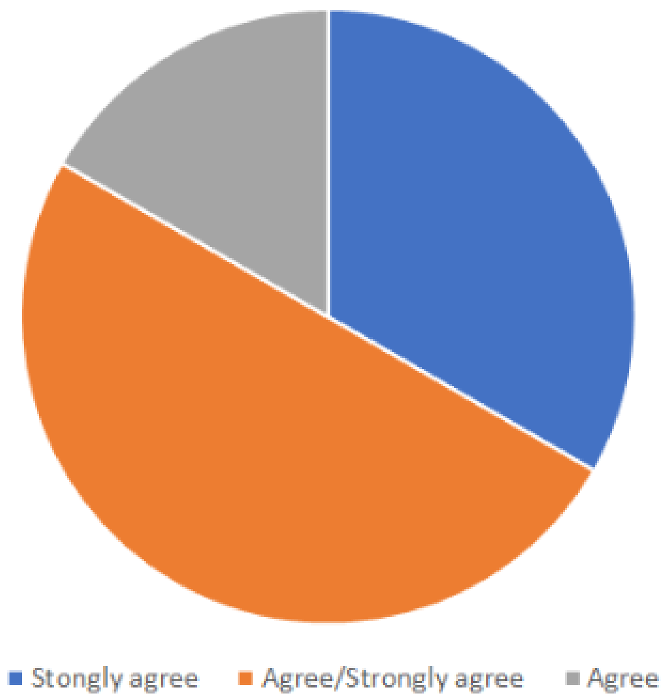


Figure 5.7: A pie chart summary for an evaluation result of the proposed solution.

6
Discussion

This chapter discusses our research findings. We refer to research results in Section 5 for discussions and summaries of the answers to our research questions. Section 6.1 and 6.2 present the implications of our research findings for research and practice, respectively. And, Secession 6.3 presents the threats to validity.

6.1 Implications for Research

6.1.1 RQ1 — Research Question 1

Recall RQ1: *What are the opportunities and challenges of the current quality overview in an Autonomous Drive Domain?*

The Autonomous Drive Domain of the case company follows a continuous development process consisting of continuous integration, validation, and verification based on an agile way of working, SAFe framework. This framework worked as a key enabler to meet the increasing demands on short development cycles and also to keep track of the quality status of the product. However, having a quality overview that could show the quality state of a product according to the needs of the different kinds of stakeholders for a particular level or end-to-end verification was lacking in the existing quality overview.

In order to pave the way towards the aim of our research study. Investigating the challenges and opportunities of the existing quality overview laid the foundation for our research study. For answering this research question, understanding of the domain and existing quality overview at each level was important for the research team. After solidifying the understanding and gaining sufficient knowledge of the domain and an existing quality overview at each level and end-to-end, an interview session was held for collecting the data. Then following the gathering of data during the interview sessions, seven themes with subsequent categories are created, as can be seen in Section 5 of the thesis report.

In the effort of eliciting the challenges and opportunities of the existing quality overview at each product level or holistic view, creating basic themes along with their inter-relation to each other is proved to be the systematic way for answering the research questions. As a result, as can be seen in Figure 5.1, themes along with their chain effect is created. The themes *verification activities* and *existing quality* *overview* serves as a foundation to explore the *challenges* and *opportunities* of existing quality overview approaches. And using the challenges, *areas to improve* theme is driven.

To precisely answer the first research question, mainly the challenges part, we have derived a hierarchy of sub-themes under the challenges that every sub-themes has a chine effect on one another. This categorizing of challenges helps to break down the issues into smaller particles so that to ease the process eliciting them and to show direction on how to resolve them. The three sub-themes created are a lack of quality information foundation, a lacking the support of quality overview life cycle, and a lacking a sufficient quality overview visualization.

6.1.2 RQ2 — Research Question 2

Recall RQ2: *What are the Key Performance Indicators (KPIs) desired by stakeholders*

In building a good quality overview of product development, Key Performance Indicators play an important role in measuring and presenting the quality of the product. As can be seen in the figure, Figure 5.1, The theme *key performance indicators* are mainly derived from the *verification activities*. However, the other themes have also a direct or indirect impact on this theme, technical infrastructure, workflows, and company context are few among them.

To precisely answer the second research question, the collected KPIs are mapped into Fishbone Diagrams[1] to clearly show the cause-effect analysis. The motivation behind using the diagram is that it is a convenient tool for a cause and effect analysis and to identify a complex interplay of causes for a specific problem or event. It is also an appropriate technique of graphical representation to explore and categorize, clearly and simply, the potential root causes of the various events or problems [39]. Hence, as the domain under study is complex and seemed to seek KPIs in a complex domain along with the intense integration of levels is intense, we believe this diagram is convenient to show the KPIs uncovered during the study.

To precisely answer the second research question, we have explored several KPIs across the integration levels and then we break them down into several categories, along with the real KPIs as subcategories, with a cause-effect approach. This helps stakeholders to perceive the KPIs in detail, and the inter-relation between in a cause-effect analyzed ways.

Furthermore, for the purpose of showing the transparency of the research results regarding the explored KPIs, we present KPIs with more information like the novelty of a particular explored KPI, the number of interviewees who mentioned it, and their respective role, as can be seen in the following list of tables (Table 6.1 - 6.8). This could provide an important sight into how we elicited the explored KPIs and

[1] https://www.moresteam.com/toolbox/fishbone-diagram.cfm.

for a possible further study in a similar domain.

The tables show the identified KPIs in different categories with the name, the number of stakeholders that mentioned the KPIs, roles of the stakeholders who mentioned the KPIs, and the novelty of the KPIs. By novelty of the KPIs, we meant to say if a particular KPI is newly discovered, already existing, or existing but in a different domain/product level than the particular product level, we take the KPI from.

No	Name	Number of Stakeholders	Roles	KPI Type
1	Turnaround time (resolution time)	2	Product owner and a CI architect.	New for some levels, yet exists in other levels
2	Test failure issue status	1	CI architect.	KPI exists across product levels.
3	Functional (Domain) requirement (does not exist)	1	CI architect.	(NEW)This KPI does not exist
4	Number of opened issues	1	solution architect.	KPIs exists levels

Table 6.1: Fault Analysis KPIs

No	Name	Number of Stakeholders	Roles	KPI Type
1	Requirement coverage via test suite (resolution time)	3	Release train engineer, product owner and system tester.	This KPIs already exists
2	Future Readiness via test suite	3	CI architect. Stakeholder (Complete level Chain) and Solution Architect	(NEW)Feature Readiness exists across various levels but feature readiness via test suite does not exists.
3	Functional (Domain) requirement (does not exist)	1	CI architect.	(NEW)This KPI does not exist
4	Legal requirements (Domain /Function level)	1	Stakeholder (complete chain)	This KPI exists in the required levels

Table 6.2: Requirement or Feature KPIs

No	Name	Number of Stakeholders	Roles	KPI Type
1	Test case automation level	1	HIL Engineer	(NEW) It does not exist in any of the levels.
2	Number of test case	2	Solution train and complete level responsible	(NEW) It does not exist in any of the levels.
3	Deliveries' status	1	Solution train engineer	(NEW) It does not exist in any of the levels.
4	Lead time between software deliveries	1	Product Owner	(NEW) It does not exist in any of the levels.

Table 6.3: Level of Automation in CI

No	Name	Number of Stakeholders	Roles	KPI Type
1	Test Case failure rate and coverage	4	Release train engineer, System Architect, CI Architect and System Tester	This KPI does not exists
2	Stability of Testing Suite	1	CI Architect	This KPI does not exists
3	Number of false positives	2	Function Developer and HIL Engineer	This KPI does not exists

Table 6.4: Test Suit (test cases quality) KPIs

No	Name	Number of Stakeholders	Roles	KPI Type
1	ECUs software downloading OTA level	1	Product Owner	This KPI exists
2	ECU Software downloading confidence level	1	Product Owner	(NEW)This KPI does not exists
3	Time duration for software downloading	1	Product Owner	(NEW)This KPI does not exists
4	Gateway's state for base tech stating	1	Product Owner	This KPI exists
5	Network Management Testing	1	Product Owner	This KPI exists

Table 6.5: Base Tech Quality KPIs

No	Name	Number of Stakeholders	Roles	KPI Type
1	ECUs software downloading OTA level	1	Product Owner	This KPI exists
2	ECU Software downloading confidence level	1	Product Owner	(NEW)This KPI does not exists
3	Time duration for software downloading	1	Product Owner	(NEW)This KPI does not exists
4	Gateway's state for base tech stating	1	Product Owner	This KPI exists
5	Network Management Testing	1	Product Owner	This KPI exists

Table 6.6: Test Environment Quality KPIs (HIL Quality)

No	Name	Number of Stakeholders	Roles	KPI Type
1	HIL self-check result status	2	HIL Engineers	This KPI currently exists
2	ECU confidence level of HIL rigs due to failure categorization	2	HIL Engineers	(NEW)This KPI does not exists
3	HIL Failure History for each product levels	1	HIL Engineers	This KPI does not exists and is desired by the stakeholders

Table 6.7: Product or ECU Quality KPIs

No	Name	Number of Stakeholders	Roles	KPI Type
1	Vehicle Project Status	1	Solution Train Engineer	New for some levels, yet exists in other levels.
2	Solution level Product KPIs	1	System Level Stakeholder	(NEW) It does not exist in any of the levels.
3	Number of failing tests Project wise	1	CI Responsible	(NEW) It does not exist project wise
4	Number of problematic feature areas Project wise	1	CI Responsible	(NEW) It does not exist project wise
5	Problem Categorization Project wise	1	CI Responsible	(NEW) it does not exist project wise
6	Project Status test results	1	Solution Train Engineer	Test results on project status is required
7	Aftermarket Customer Satisfaction	1	AfterMarket	This KPI does not exists and is required by the stakeholder

Table 6.8: High Level Overview KPIs

6.1.3 RQ3 — Research Question 3

Recall RQ3 - *In what ways should the stakeholders prefer a quality overview to get visualized?*

This research question acts as an enabler for the quality overview tool, along with the key performance indicators. To answer the research question, we depend on the information gained from interviews and other explored themes like challenges, areas to improve, verification activities, and existing quality overview opportunities. This research question helped in the visualization of the quality status which serves as an important section for the good quality overview of the system.

To precisely answer the third research question, it was important to keep in mind the important features that need to be visualized. Certain sub-themes were created representing the various contexts through which stakeholders could visualize the quality status of the product in their respective product levels and others according to their needs. Hence, we have categorized the various sub-themes as, bird's view, technical features or products, release quality overview, adaption per stakeholder, and activity enabling visualization.

The reason behind bringing out the Bird's view as one of the sub-theme is we got to know that this sub-theme could help stakeholders especially at high levels and of a strategic domain to track the status from a high-level view perspective. The sub-theme for technical features or products shows various features desired by the stakeholders which they would like to visualize for the quality overview of their product. This sub-theme would help in providing the technical details for the product across certain verification levels. We have categorized the release quality overview as one of the sub-theme because it contains all those preferences which direct towards the release quality overview. The other sub-themes are adaption per stakeholder and activity enabling visualization which contain those preferences which help stakeholders to adopt their own way of visualizing quality information state and as an activity enabler respectively.

Furthermore, for the purpose of showing the transparency of the research results regarding the explored visualization preferences, we present visualization preferences with more information like the novelty of a particular explored visualization preference, the number of interviewees who mentioned it, and their respective role, as can be seen in the following list of tables (Table 6.9 - 6.13). This could provide an important sight into how we elicited the explored visualization preferences and for a possible further study in a similar domain.

The tables show the identified visualization preferences in different categories with the name, number, and roles of stakeholders who mentioned the visualization preferences, and the novelty of the visualization preferences. By novelty of the visualization preferences, we meant to say if a particular visualization preference is newly discovered, already existing or existing but in a different domain/product level than the particular product level, we take the visualization preference from.

No	Name	Number of Stakeholders	Roles	KPI Type
1	Visualization of test results in an abstraction level	2	Domain product owner and CI architect	(NEW) It does not exist in any of the levels.
2	Products failure trends via graph	2	CI architects.	(NEW) It does not exist in any of the levels.
3	Color coding for test cases	1	CI architect.	(NEW) It does not exist in any of the levels
4	A visualization for stakeholders to follow the quality of any products and environment settings	4	HIL engineers and Product Owner (HIL)	(NEW) It does not exist in any of the levels.

Table 6.9: Technical features visualization preferences

No	Name	Number of Stakeholders	Roles	KPI Type
1	Releases' Visualization	2	System tester and Release train engineer	(NEW) It does not exist in any of the levels.
2	Test result details for every level	8	CI architect, product owner, release train engineer, system architect, HIL engineers and solution architects.	(NEW) It does not exist in any of the levels.
3	Tracking releases' test failures issues	2	System Tester and Release Train Engineer	This visualization preference already exists
4	Test suite result difference between releases.	1	CI Architect	(NEW) This visualization preferences does not exists in any of the product levels.

Table 6.10: Release quality overview visualization preferences

No	Name	Number of Stakeholders	Roles	KPI Type
1	Test environment readiness	4	HIL Engineers and Product Owner	(NEW) This VP does not exists and is discovered
2	Test environment Failure History	4	HIL Engineers	NEW)This preference does not exist in the required level but stakeholders would like to have it, this preference was explored.
3	Real-time test results from environment for SUT	3	HIL Engineers	(NEW)This preference is identified and it does not exist in any of the levels.

Table 6.11: Activity enabling information visualization preferences

No	Name	Number of Stakeholders	Roles	KPI Type
1	Independent Visualization	3	CI Architects and HIL Engineer	(NEW)This preference does not exists in any of the level and is discovered
2	Simplicity	1	CI Architect	(NEW)This preference does not exists in all levels except few and is explored
3	Customizable Visualization	2	CI Architects	(NEW)This preference does not exists in any of the level and is discovered

Table 6.12: Adaption per stakeholders visualization preferences

No	Name	Number of Stakeholders	Roles	KPI Type
1	Bird's view of all verification activities across product levels.	1	Stakeholder up to complete level	(NEW) This preference does not exist across levels and stakeholders would like to have it across levels.
2	A visualization for both manual and automated testing strategy and corresponding results.	3	CI Architects and Complete level stakeholder	(NEW) Manual testing are not visualized enough the same as automated testing.
3	A visualization for both legal requirement	2	System Architect Complete level stakeholder	(NEW) This preference does not exists in any product level.

Table 6.13: Bird's view of all verification activities and results

6.2 Implications for Practice

Our results allow practitioners to implement a quality overview tool equipped with KPIs and visualization preferences for stakeholders. The result also could provide direction on which the case-company revise the ways of working and overall verification strategy. As we get into the depth exploration of the case company, the results could give a deeper insight into the current challenges, opportunities, KPIs, and visualization preference so that the management could work on the improvement plans suggested in the study.

Our findings on KPIs and visualization preferences as enablers for quality overview tools can help to get an understanding of the relevant quality indicators and presentations that provides a convenient quality status of products at an integration level. Practitioners can use this study as a starting point to reconsider the existing and the newly uncovered KPIs and visualization preferences, in their daily product development and verification activities, to consolidate the presentation of the quality status of products.

Our proposed alternative tool can be used as a prototype, that shall be consolidated by further study and later used as a prior design, for the real implementation of the estimated quality overview tool.

6.3 Threats to Validity

This section aims to discuss some observed threats to validity during the thesis work and the strategies applied to mitigate them. The following categories of validity threats are adopted from a study by Runeson and Höst [13].

6.3.1 Construct validity

When we met stakeholders for an interview and elaborate on our research questions, it was always cumbersome to interpret the research objectives in the same way as ours, which could be a threat to construct validity. Hence, to mitigate this threat, we gave an introduction to the research objectives supported by the graphical representation of the research scope gathered from the domain exploration in the case company. Moreover, we also try to present our preliminary findings and literature review results when we feel some misunderstandings occur with our interviewees.

During the data analysis industrial workshop, for the purpose of creating the quality overview model, participants from the case-company were involved. This could be a threat to construct validity, lacking to accommodate an academic perspective. Hence, to mitigate this problem, a participant from academia was involved in the workshop.

6.3.2 Internal validity

All participants of the proposed solution evaluation had knowledge about the model which eventually mapped into the proposed solution. Hence, their previous interaction with the model could affect the evaluation result and subsequently become a source of internal validity threat. To partially mitigate this problem, the expert-review evaluation questions were made to assess the functionality with Likert-scale closed questions, so that the participants would only rank the proposed tool for functional fulfillment.

6.3.3 External validity

During the analysis of external validity, we try to analyze to what extent the findings are of relevance for other cases. As the thesis was conducted at a case company, and case studies generally result in findings with lower generalizability. This can have a negative effect on the generalization of the results. However, to partially mitigate this problem and increase generalizability, we tried to involve participants from academia during drawing the thesis result of the thesis work, where the results could extend to cases (other complex domains) that have common characteristics and hence for which the finding are relevant. Literature also indicates that many companies within the automotive or autonomous drive domain have a lot in common with the case company, Volvo Cars.

6.3.4 Reliability validity

The interview guide tool was developed after the domain exploration and literature reviews were done. This can pose a threat to the reliability of the research findings validity as different researchers can develop interview guides in different ways.

However, to partially mitigate this problem, we put the interview guide through the review of participants from both the industry and academia perspective.

6.4 Future Work

There are certain interesting routes and promising possibilities that could be explored for future work in this area. This section presents some of them.
The first aspect would be on how to proceed with the exploring and working on the implementation of quality overview tool solutions according to the proposed prototype and further exploration. It will be interesting to implement the tool and study their acceptance in practice. Moreover, the proposed prototype shall also be evolved through a reasonable iteration of user testing.

Another interesting concept is KPIs, and investigating how it could ignite/trigger bad behavior in developers while achieving the best quality level required. Despite the important role KPIs play in ensuring a good quality overview of products, some of the KPIs could ignite positive or negative behavior among the developers or stakeholders engaged in the product development and verification. This argument can be explained further by the following excerpt from our interview whose responsibility is the product owner at a component level.

"*Suppose there is a KPI that shows the delivery time of the product based on the timesheet of developers. Due to this KPI, some developers might focus more on their daily time reports in spite of focusing on the quality of the product no matter what, the timesheet could hinder their productivity for that specific task. So, this scenario could ignite the negative behavior among developers which would ultimately affect the quality of the product.*"

– Product Owner/Component level

Therefore, investigation of negative traits of the KPIs might also be needed in the future work.

7
Conclusion

The role of quality in the complex domain especially in the autonomous domain is as important as the production because an autonomous car is highly dependent on error-free software, and the issue of proven quality assurance becomes increasingly urgent [16]. Although stakeholders of different product levels are capable enough to understand the quality state of their product, having a good quality overview that could provide the understanding and status of the product in an integration and end-end level is still quite challenging in this domain. As the case company is also undergoing a transformation of SAFe which is based on continuous integration, continuous development, and continuous verification and validation, and thus the need for understanding the quality of the product for guiding the development efforts is continuously needed. Therefore, having a good quality overview at each product level and end-to-end is important.

In this thesis, we explored the existing quality overview, its challenges, opportunities, key performance indicators, and visualization preferences. As we applied an exploratory case study, we conducted 16 interviews with stakeholders from different product levels of the case company.

The result has shown that the domain introduces certain challenges with respect to quality overview. Despite some tools already available in the domain that have certain opportunities (also explored during the study), a good quality overview enabler, that has valuable key performance indicators along with stakeholder's interest centered visualization preferences, is demanded. As per our exploration across similar studies in similar domains, such a tool has never been studied before. Hence, we have identified important key performance indicators and visualization preference, as can be seen in the result section, Section 5.

To validate the study results, we have also developed a solution and subsequently conducted a follow-up evaluation. As a result of expert-review evaluation indicates in Section 5.5.2, the proposed solution is successfully accounted for a candidate alternative solution.

Bibliography

[1] Dajsuren, Yanja, et al. "Automotive ADLs: a study on enforcing consistency through multiple architectural levels." Proceedings of the 8th international ACM SIGSOFT conference on Quality of Software Architectures. ACM, 2012.

[2] Jacob, Stacy A., and S. Paige Furgerson. "Writing Interview Protocols and Conducting Interviews: Tips for Students New to the Field of Qualitative Research." Qualitative Report 17 (2012): 1-10. Web.

[3] Nilsson, Agneta, Bosch, Jan, and Berger, Christian. (2014): 171-86. Print.

[4] Seaman, C.B. "Qualitative Methods in Empirical Studies of Software Engineering." IEEE Transactions on Software Engineering 25.4 (1999): 557-72. Web.

[5] Creswell, John W. Research Design : Qualitative, Quantitative, and Mixed Methods Approaches. 3.rd ed. Thousand Oaks, Calif.: Sage, 2009. Print.

[6] Karlstro, D. (2006). Integrating agile software development into stage-gate managed product development, 203–225. https://doi.org/10.1007/s10664-006-6402-8

[7] Borg, Markus, Per Runeson, and Anders Ardö. "Recovering from a Decade: A Systematic Mapping of Information Retrieval Approaches to Software Traceability." Empirical Software Engineering 19.6 (2014): 1565-616. Web.

[8] Deemer, Pete, et al. "The scrum primer V 1.2." Scrum Alliance (2010).

[9] Johannessen, Per, Öjvind Halonen, and Ola Örsmark. "Functional safety extensions to automotive SPICE according to ISO 26262." International Conference on Software Process Improvement and Capability Determination. Springer, Berlin, Heidelberg, 2011.

[10] Pelliccione, Knauss, Heldal, Magnus Ågren, Mallozzi, Alminger, and Borgentun. "Automotive Architecture Framework: The Experience of Volvo Cars." Journal of Systems Architecture 77 (2017): 83-100. Web.

[11] Myers, Michael D., and Michael Newman. "The qualitative interview in IS research: Examining the craft." Information and organization 17.1 (2007): 2-26.

[12] Seaman, Carolyn B. "Qualitative methods in empirical studies of software engineering." IEEE Transactions on software engineering 25.4 (1999): 557-572.

[13] Runeson, Per, and Martin Höst. "Guidelines for Conducting and Reporting Case Study Research in Software Engineering." Empirical Software Engineering 14.2 (2009): 131-64. Web.

[14] Kasauli, Rashidah, et al. "Requirements engineering challenges in large-scale agile system development." 2017 IEEE 25th International Requirements Engineering Conference (RE). IEEE, 2017.

[15] Anzul, Margaret, et al. Doing qualitative research: Circles within circles. Routledge, 2003.

[16] Bagloee, S.A., Tavana, M., Asadi, M. et al. Autonomous vehicles: challenges, opportunities, and future implications for transportation policies. J. Mod. Transport. 24, 284–303 (2016). https://doi.org/10.1007/s40534-016-0117-3

[17] Nilsson, Agneta, Jan Bosch, and Christian Berger. "Visualizing testing activities to support continuous integration: A multiple case study." International Conference on Agile Software Development. Springer, Cham, 2014.

[18] Nilsson, Agneta, Jan Bosch, and Christian Berger. "The CIViT model in a nutshell: Visualizing testing activities to support continuous integration." Continuous software engineering. Springer, Cham, 2014. 97-106.

[19] Stuckey, Heather. "The Second Step in Data Analysis: Coding Qualitative Research Data." 3.1 (2015): 007-10. Web.

[20] Plas, Jeanne M., Steinar Kvale, and STEINAR AUTOR KVALE. Interviews: An introduction to qualitative research interviewing . Sage, 1996.

[21] Shen, Jing. "Investigation of how to implement successful KPIs for organizations–based on an empirical study at an international organization." (2013).

[22] Wohlrab, Rebekka, Steghöfer, Jan-Philipp, Knauss, Eric, Maro, Salome, and Anjorin, Anthony. (2016): Proceedings Of 24th IEEE International Requirements Engineering Conference (RE' 16), 2016. Print.

[23] Olsson, Thomas, and Krzysztof Wnuk. "QREME–Quality requirements management model for supporting decision-making." International Working Conference on Requirements Engineering: Foundation for Software Quality. Springer, Cham, 2018.

[24] Gardner, Lyn. "Review: Memoing in Qualitative Research: Probing Data and Processes." Journal of Research in Nursing 13.1 (2008): 76-77. Web.

[25] Walker, Miriam, Leila Takayama, and James A. Landay. "High-fidelity or low-fidelity, paper or computer? Choosing attributes when testing web prototypes." Proceedings of the human factors and ergonomics society annual meeting. Vol. 46. No. 5. Sage CA: Los Angeles, CA: SAGE Publications, 2002.

[26] Johansson, Maria, and Mattias Arvola. "A case study of how user interface sketches, scenarios and computer prototypes structure stakeholder meetings." Proceedings of the 21st British HCI Group Annual Conference on People and Computers: HCI... but not as we know it-Volume 1. British Computer Society, 2007.

[27] Liggesmeyer, Peter, et al. "Quality improvement through visualization of software and systems." Quality Assurance and Management, InTech (2012): 315-334.

[28] Robson, Colin. "Real world research 2nd edition." Malden: BLACKWELL Publishing (2002).

[29] Yin, Robert K. "Case Study Research Design and Methods, 5th Revise." (2013).

[30] Benbasat, Izak, David K. Goldstein, and Melissa Mead. "The case research strategy in studies of information systems." MIS quarterly (1987): 369-386.

[31] Wohlin, Claes, Martin Höst, and Kennet Henningsson. "Empirical research methods in software engineering." Empirical methods and studies in software engineering . Springer, Berlin, Heidelberg, 2003. 7-23.

[32] Easterbrook S., Singer J., Storey MA., Damian D. (2008) Selecting Empirical Methods for Software Engineering Research. In: Shull F., Singer J., Sjøberg D.I.K. (eds) Guide to Advanced Empirical Software Engineering. Springer, London

[33] Buzan, Tony. Use Both Sides of Your Brain: New Mind-mapping Techniques to Help You Raise All Levels of Your Intelligence and Creativity based on the Latest Discoveries about the Human Brain. Penguin Books USA Incorporated, 1989.

[34] Policy, Cloud Computing. "Guideline 3." (2013).

[35] Myers, Michael D., and Michael Newman. "The qualitative interview in IS research: Examining the craft." Information and organization 17.1 (2007): 2-26.

[36] Stolberg, Sean. "Enabling agile testing through continuous integration." 2009 Agile Conference. IEEE, 2009.

[37] Alsaqaf, Wasim, Maya Daneva, and Roel Wieringa. "Quality requirements challenges in the context of large-scale distributed agile: An empirical study." Information and software technology 110 (2019): 39-55.

[38] Babich, Nick. "Prototyping 101: The Difference between Low-Fidelity and High-Fidelity Prototypes and When to Use Each." (2017).

[39] Coccia, Mario. "The Fishbone diagram to identify, systematize and analyze the sources of general purpose Technologies." Journal of Social and Administrative Sciences 4.4 (2018): 291-303.

A
Appendix 1

A.1 Interview guide tool

Interview Guide

Introduction

1. Present ourselves.
2. Present interview goals.
   - Identify opportunities and challenges with the current ways of quality overview (and visualizing it) for the product in the autonomous drive domain.
   - Identify key performance indicators desired by stakeholders from the verification activities used during the complete development flow, from software module update to complete car.
   - Understand stakeholders' preferred quality overview on their respective scope/levels and for the end-to-end verification.
3. Data will be used anonymized in our master thesis to report on the quality overview visualization, and possibly in a follow-up research papers.
4. Your name and the company's name will not be associated with your answers.
5. Can we record?
   - During the interview session, we will clarify that the interview contents will not leave VCC. We will only listen to the recording and transcribe it as an input for the study.

Interview Questions

Main research questions:

RQ1: What are the opportunities and challenges of quality overview (and visualization) in the autonomous drive domain?

RQ2: What are the Key Performance Indicators (KPI) desired by stakeholders?

RQ3: In what ways would the stakeholders prefer a quality overview?

Questions on the background of interviewee:

1. What is your role in the company?
   - How long have you been in this role?
   - What did you do prior to this role?
2. In which other areas have you worked previously?
3. How long have you worked in the automotive industry, overall?

Questions related to opportunities and challenges of quality overview (RQ1):

1. ***RQ1-Q1:*** Which role in the following list best matches your current role at Volvo Cars?
    a. Software module tester
    b. Developer
    c. Component tester
    d. Combination responsible
    e. System/completer level tester
    f. Upper management
    g. Or, other?

2. ***RQ1-Q2:*** In which level of the system development are you working now (or which level in the Volare, continuous integration flow, are you working now?
    a. Unit level (Software module test level)
    b. ECU level
    c. Domain level
    d. System level
    e. Complete Car level

3. ***RQ1-Q3:*** What verification activities happen at your level?

4. ***RQ1-Q4:*** How do you currently get the information you need from the verification activities (and result) that are of importance to you?

[***Here, we will give a short introduction to what we mean by "Quality overview".***]

5. ***RQ1-Q5:*** What does "Quality Overview" mean in your role and scope of working? (What methods, if any, do you currently use to get a quality overview to help your work and decision making?)

6. ***RQ1-Q6:*** (How) would such a quality overview provide value to your daily work? (And, What opportunities have you found for improving these methods?)
    [--> Then, this will yield the opportunities of a good quality overview.]
    - Aspects related to verification (verification activities).
    - Aspects related to continuous integration.
    - Aspects related to visualization.
    - Aspects related to input from other roles/scopes/levels.

7. ***RQ1-Q7:*** Is the current quality overview sufficient? Elaborate, especially where not. (Or, What do you see as the main challenges for getting a quality overview based on the verification activities that are relevant to you in the current quality overview?)
    - Aspects related to verification (verification activities).
    - Aspects related to continuous integration.
    - Aspects related to visualization.
    - Aspects related to input from other roles/scopes/levels.

8. ***RQ1-Q8:*** Do you feel sufficiently informed about verification activities or quality overview on integration levels other than your primary one?
    [If you do, what opportunities and challenges have you observed?]

9. ***RQ1-Q9:*** Do you have an existing end-to-end quality overview methods for the all verification activities from software module into complete Car level?
    [If applicable, what opportunities for improvement would you identify? And Challenges?]

10. ***RQ1-Q10:*** Based on what you just said above, would you agree if we conclude that enabling a quality overview during a continuous development with an end-to-end verification support the complex product development using an agile development (such as SAFe)?
    [If Yes, How? E.g. in decision making activities (regarding to improvement plan), quick feedback loops and plan-driven frequent releases.]

Questions related to desired Key Performance Indicators (KPIs, RQ2):

1. ***RQ2-Q1:*** How do you know (or measure) if the available verification process (testing strategy) in your scope/level is good enough to verify the product?
    [E.g. How do you measure if there are sufficient testing activities.]

2. ***RQ2-Q2:*** What kind of KPIs do you get or calculate, currently, in your primary level (scope)?
    [**Optional or plan for follow-up interviews:**
    - Do you have a new KPIs, in mind, to add on top of the existing ones?

- Do you have a certain thresholds, in mind, for the above KPIs?
- Do you also happen to know, or have in mind, a certain KPIs that you would suggest for verification activities in other integration levels/scopes?]

3. ***RQ2-Q4:*** Do you think a quality overview for all integration levels with a desired KPIs of the verification activities, going towards CI, support the continuous development for complex products? If Yes, would you add more points on the following questions?
    - Do you have some KPIs, in your mind, which could be common for two or more scopes/levels.
    - Can you suggest your own KPIs in the complete overview (end-to-end overview), if you have any?

Questions related to preferred quality overview (RQ3):

1. ***RQ3-Q1:*** How do you describe the current way of seeing a quality information from the quality overview of verification activities within your scope/ level? [E.g. Visualization methods]

2. ***RQ3-Q2:*** If you were to adopt a quality overview along with a valuable KPIs, in what ways would you prefer, as one of the stakeholders in your scope/level, for you to see the quality overview?

3. ***RQ3-Q3:*** In an end-to-end quality overview, in what format would you like to see the quality information in scopes/ levels other than your primary one? [Elaborate? You might also elaborate it using your previous experience]

Inspirational Questions:

1. ***IQ-Q1:*** If we ignore complexities and the resources required, what would your ideal quality overview look like?

2. ***IQ-Q2:*** How do you see quality overview working out five years from now?

Feedback:

1. Is there anything we forgot to ask about that would help us in our study?
2. We appreciate the time you took for this interview. Do you have any recommendations for who else we should talk to?

A.2 Expert evaluation questionnaires

Expert-Review Question

1. The proposed quality overview solution seems to give quality state of product in an efficient way?
   - Strongly agree
   - Agree
   - Neither agree nor disagree
   - Disagree
   - Strongly disagree

   Comment:

2. The proposed solution seems realistic?
   - Strongly agree
   - Agree
   - Neither agree nor disagree
   - Disagree
   - Strongly disagree

   Comment:

3. The proposed solution should be implemented soon at VCC?
   - Strongly agree
   - Agree
   - Neither agree nor disagree
   - Disagree
   - Strongly disagree

   Comment:

4. The proposed quality overview helps for stakeholders on their decision-making activities?
   - Strongly agree
   - Agree
   - Neither agree nor disagree
   - Disagree
   - Strongly disagree

   Comment:

5. The proposed quality overview gives end-to-end quality overview?
   - Strongly agree
   - Agree
   - Neither agree nor disagree
   - Disagree

- Strongly disagree

Comment:

6. The proposed quality overview contributes on informing product failures trends so that stakeholders could make an improvement plan?
    - Strongly agree
    - Agree
    - Neither agree nor disagree
    - Disagree
    - Strongly disagree

    Comment:

7. The proposed solution creates a good quality overview of products for different stakeholders' perspective at Active safety?
    - Strongly agree
    - Agree
    - Neither agree nor disagree
    - Disagree
    - Strongly disagree

    Comment:

8. The challenges, areas to improve, KPIs and visualization preferences explored in the study are incorporated in the proposed quality overview solution?
    - Strongly agree
    - Agree
    - Neither agree nor disagree
    - Disagree
    - Strongly disagree

    Comment:

9. The proposed quality overview is an important enabler for large scale agile way of working? [Frequent delivery with quick feedback and data driven decision making]
    - Strongly agree
    - Agree
    - Neither agree nor disagree
    - Disagree
    - Strongly disagree

    Comment:

10. The proposed quality overview solution allows stakeholders to follow the product quality through the different integration levels?
    - Strongly agree
    - Agree
    - Neither agree nor disagree
    - Disagree

- Strongly disagree

Comment:

11. Did the proposed quality overview solution meet your expectations about the kind of quality overview you wish to see as per the completion of this study?
    - Strongly agree
    - Agree
    - Neither agree nor disagree
    - Disagree
    - Strongly disagree

    Comment:

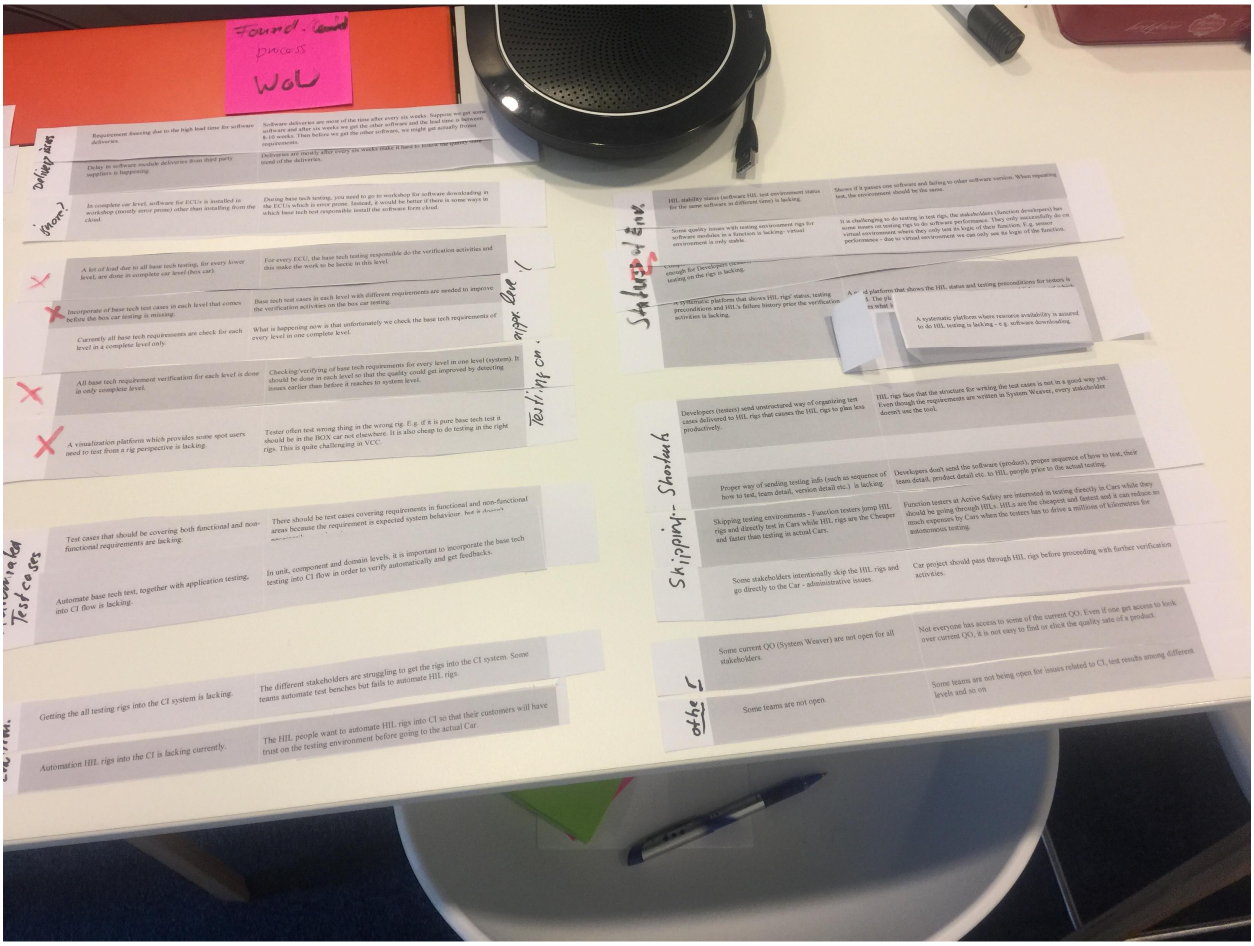
12. Does the proposed quality overview solution meet the scenarios defined below?
    - Strongly agree
    - Agree
    - Neither agree nor disagree
    - Disagree
    - Strongly disagree

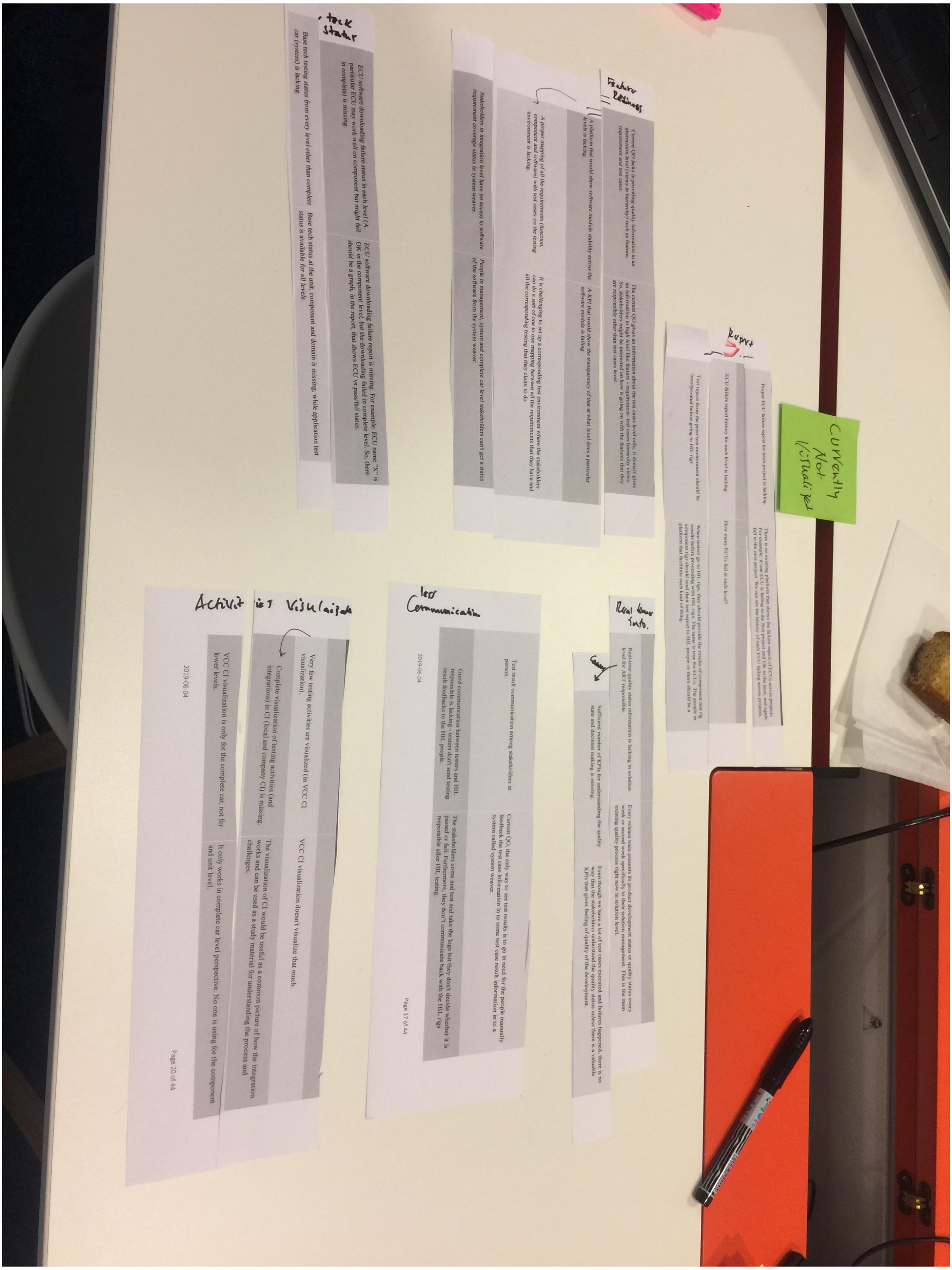
    Comment:

B

Appendix 2

B.1 Transcripts' codes

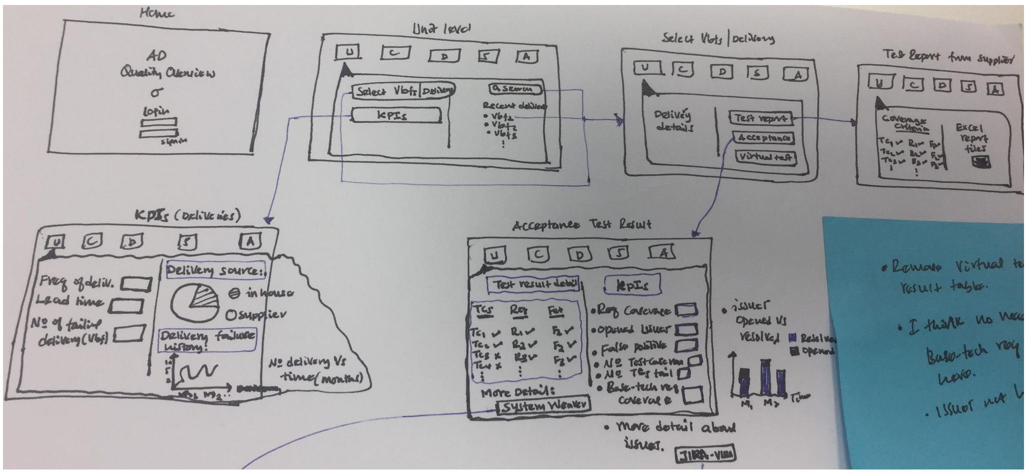


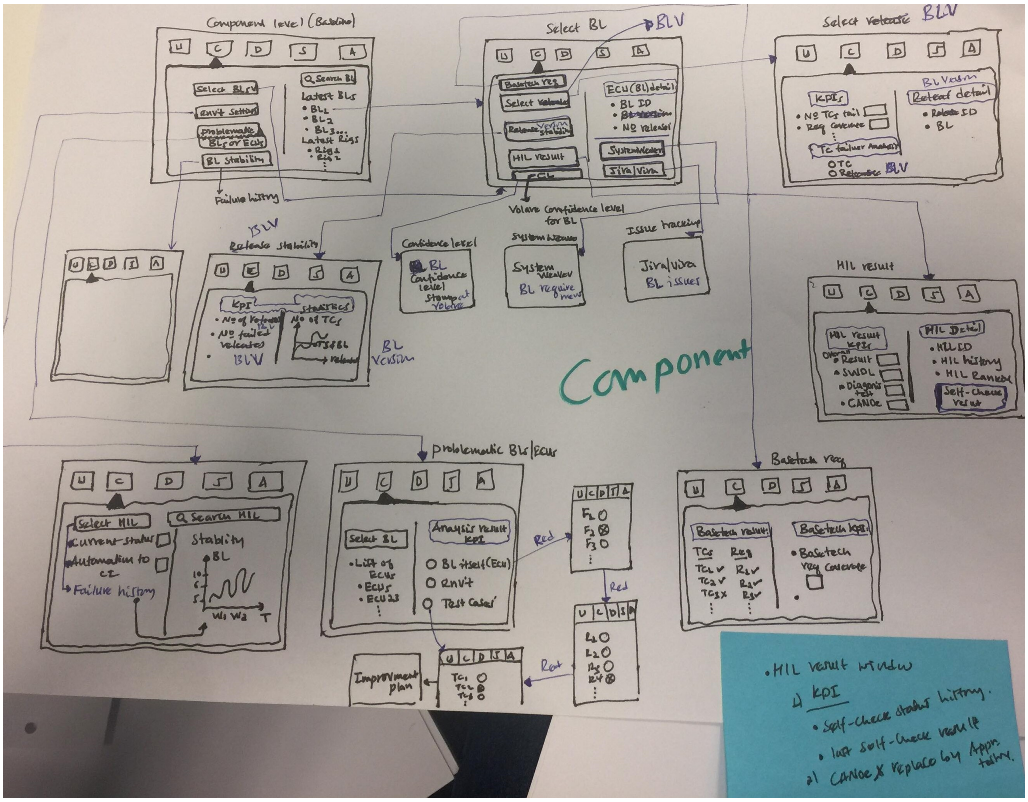

Currently Not Visualized

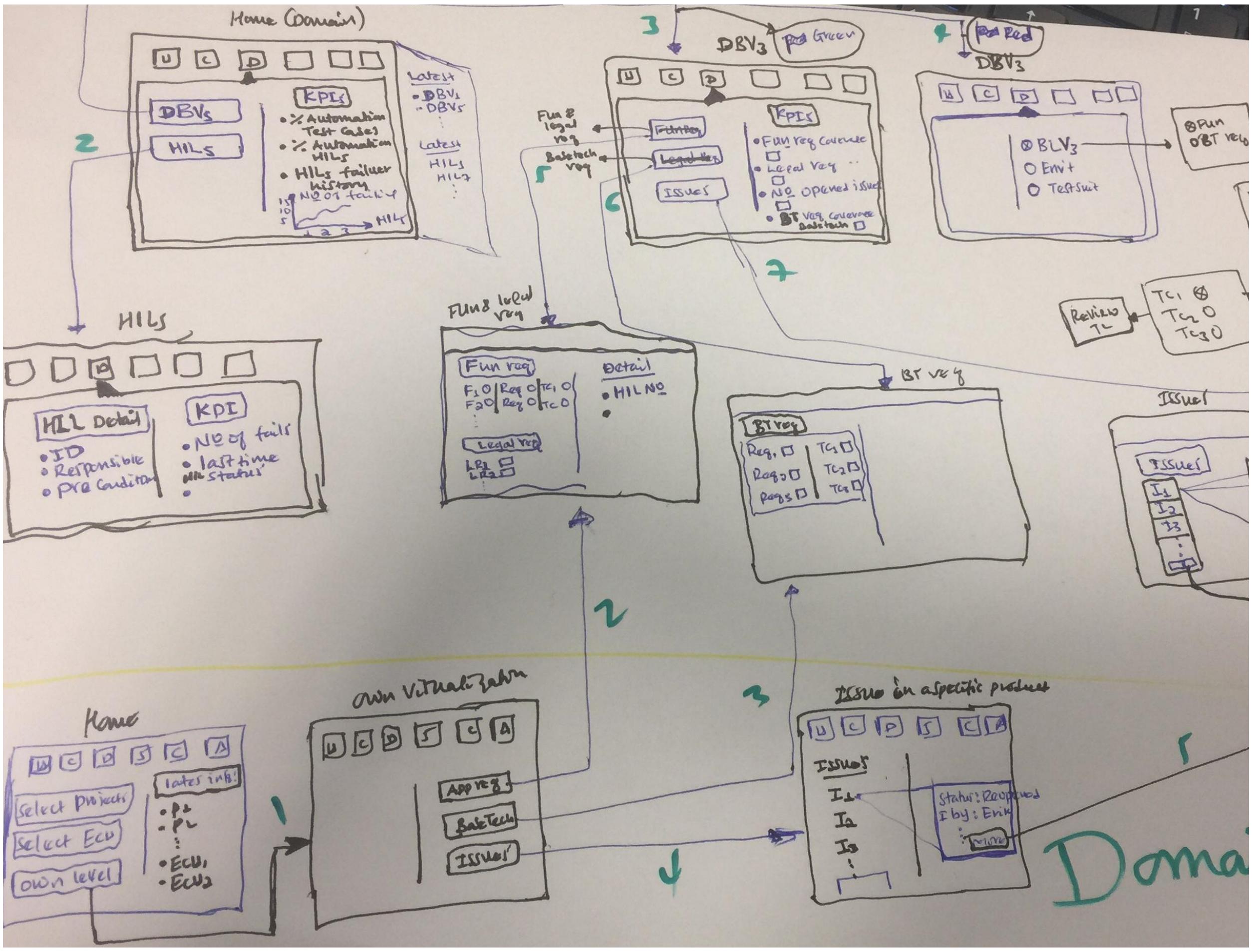
C

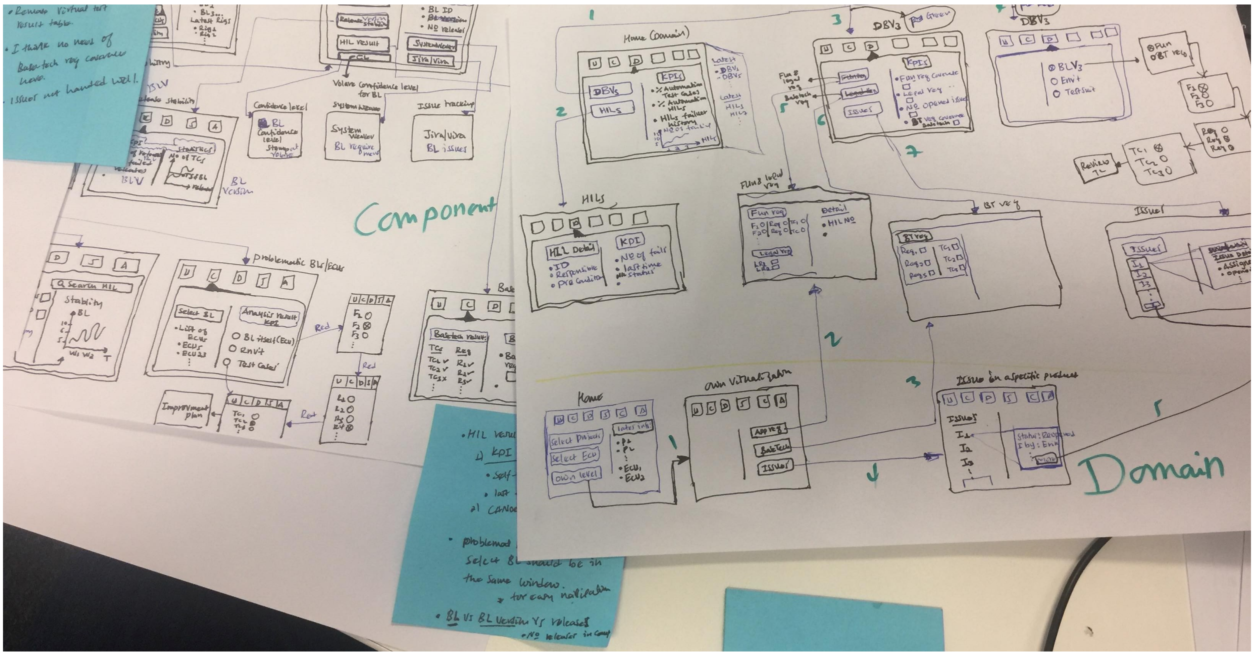
Appendix 3

C.1 Low-fidelity prototype

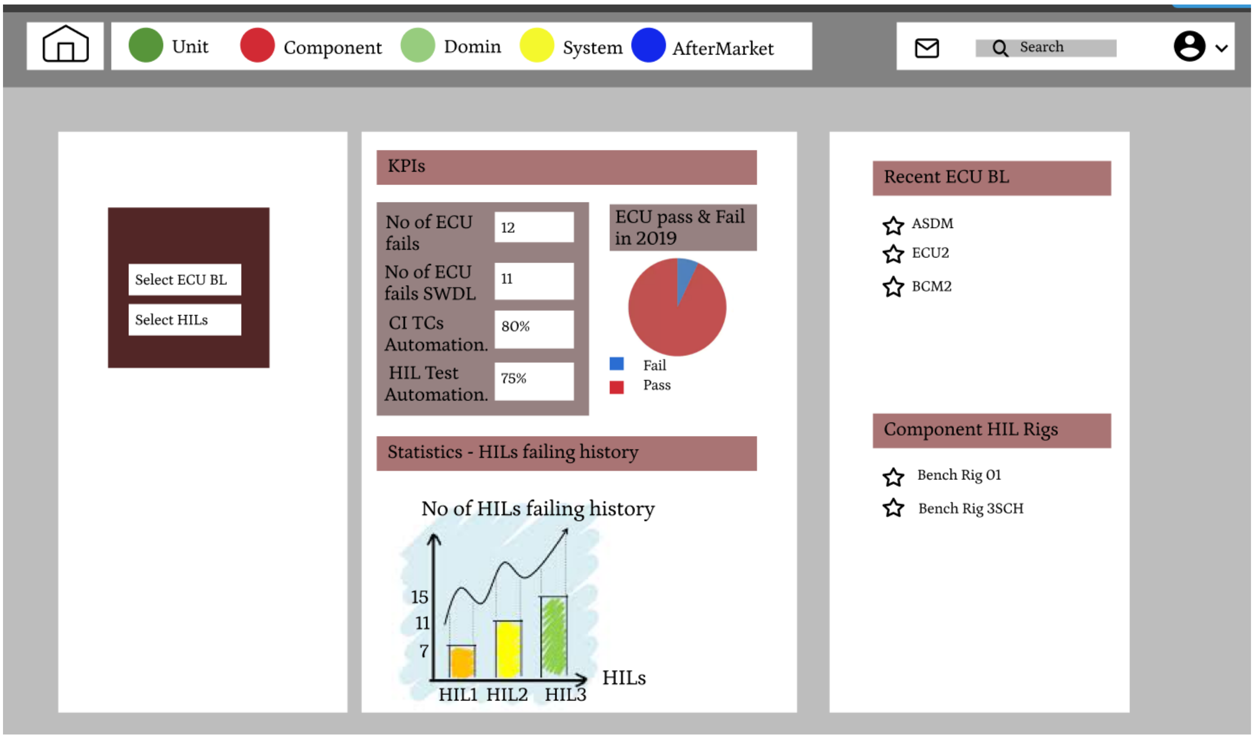


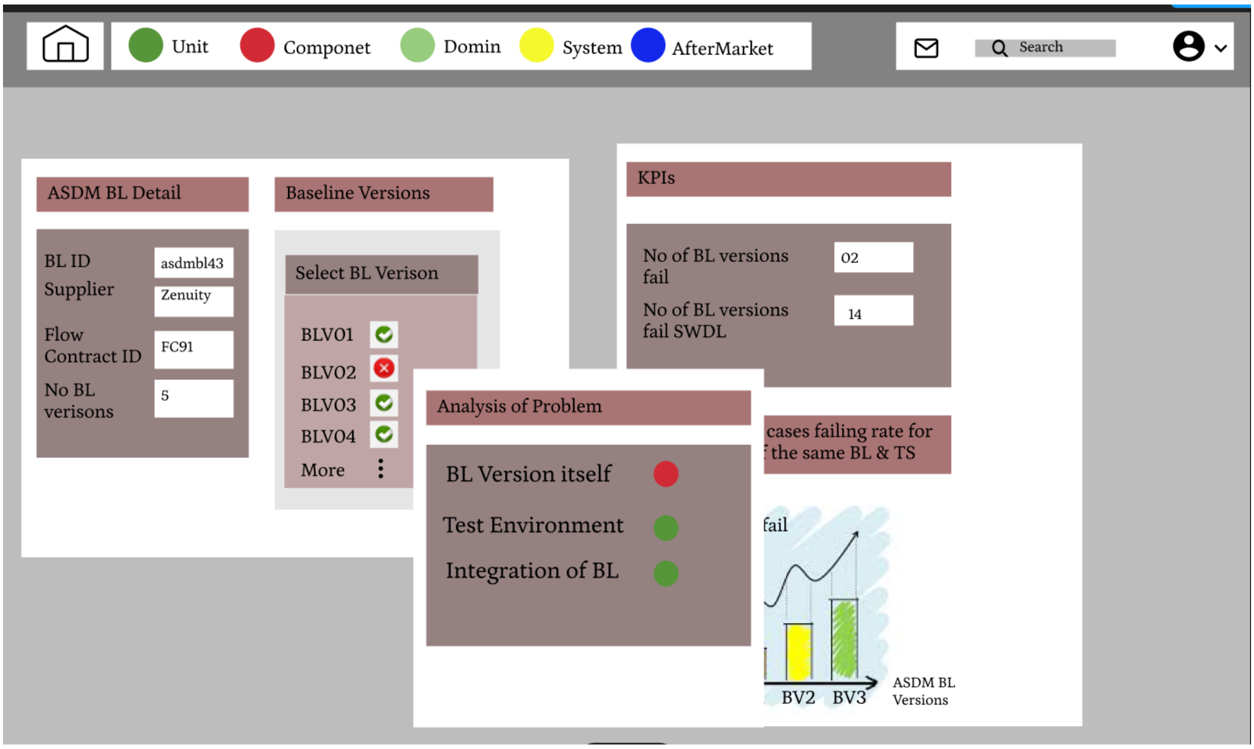



Home (Domain)
DBVs
HILs
KPIs
HIL Detail
KPI
Fun req
Detail
Legal req
BT req
Issues
own Virtualization
Home
Select Projects
Select Ecu
own level
Issue in a specific product
Domain

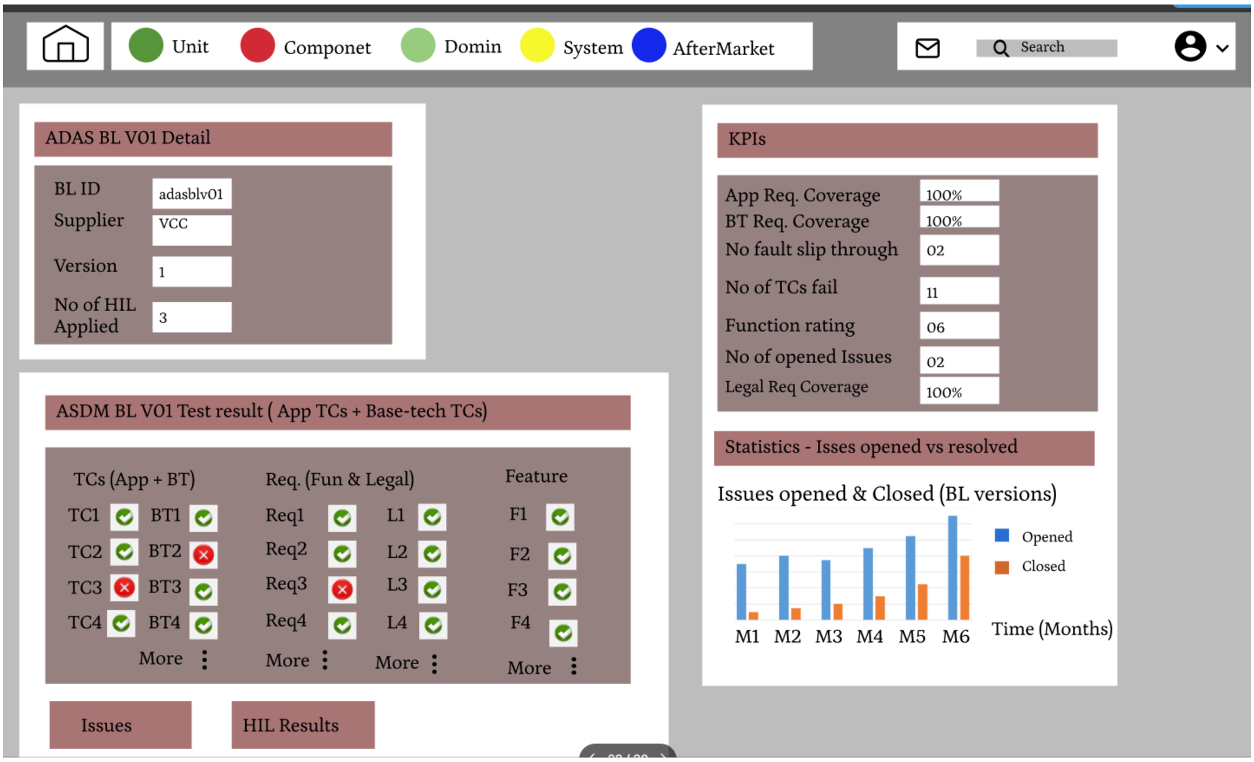

Component
Domain

C.2 High-fidelity prototype





Unit
Componet
Domin
System
AfterMarket
Search
ASDM BL Detail
BL ID
asdmbl43
Supplier
Zenuity
Flow Contract ID
FC91
No BL verisons
5
Baseline Versions
Select BL Verison
BLV01
BLV02
BLV03
BLV04
More
KPIs
No of BL versions fail
02
No of BL versions fail SWDL
14
cases failing rate for
the same BL & TS
Analysis of Problem
BL Version itself
Test Environment
Integration of BL
fail
BV2
BV3
ASDM BL Versions



Unit
Componet
Domin
System
AfterMarket
Search
ADAS BL V01 Detail
BL ID
adasblv01
Supplier
VCC
Version
1
No of HIL Applied
3
KPIs
App Req. Coverage
100%
BT Req. Coverage
100%
No fault slip through
02
No of TCs fail
11
Function rating
06
No of opened Issues
02
Legal Req Coverage
100%
ASDM BL V01 Test result (App TCs + Base-tech TCs)
TCs (App + BT)
Req. (Fun & Legal)
Feature
TC1
BT1
Req1
L1
F1
TC2
BT2
Req2
L2
F2
TC3
BT3
Req3
L3
F3
TC4
BT4
Req4
L4
F4
More
More
More
More
Issues
HIL Results
Statistics - Isses opened vs resolved
Issues opened & Closed (BL versions)
Opened
Closed
M1
M2
M3
M4
M5
M6
Time (Months)

D
Appendix 5

D.1 Volare