Bachelor Degree Project

Integration Testing of Electronic Control Units for Heavy Vehicles

Author: Noorie Esmaili, Oyejobi Ibrahim Olamide
Supervisor: Faiz ul Muram
Examiner: Hemant Gheyvat
Semester: VT 2023
Subject: Computer Science
Abstract

In today’s complex landscape of embedded software systems, the robust integration testing need is more critical than ever. This bachelor’s thesis project delves into the integration testing of an articulated hauler dumping system for Volvo Construction Equipment (Volvo CE). The study focuses on the architecture, functional and safety requirements and overall performance of the system. The complexity of modern construction machinery, driven by sophisticated Electronic Control Units (ECUs), demands precise coordination and flawless operation. Inaccuracies or failures in integrated ECUs can result in operational inefficiencies, safety risks, and substantial downtime costs. This paper offers valuable insights into a systematic approach designed to tackle these challenges head-on within the domain of articulated hauler dumping systems. Our exploration comprehensiveness employed diverse testing methodologies, encompassing scenario-based, boundary and stress testing, integration testing, and user interaction testing. This multifaceted approach enabled a thorough understanding of the system’s behavior, reliability, and responsiveness. The research aims to clarify the importance of comprehensive integration testing and its role in ensuring optimal performance, safety, and complex embedded systems efficiency. This study is a valuable resource for researchers, engineers, and professionals navigating the complexities of embedded systems integration, offering actionable insights and strategies for achieving robustness and reliability in intricate machinery operations.

Keywords: Integration testing, Articulated hauler dumping system, Embedded software, Software architecture
Preface

Undertaking this degree project has been a remarkable journey that has allowed us to delve into the integration testing domain for complex software systems. We are grateful to Volvo CE for the opportunity to undertake this thesis project.

We will extend our sincere gratitude to Faiz ul Muram, our university supervisor, whose guidance, expertise, and constructive feedback have been instrumental in shaping the trajectory of this research. Your insights have been invaluable, and your constant encouragement has motivated us to strive for excellence.

We sincerely appreciate Rikard Friberg and Ove Welander, Volvo CE Software Engineers, and supervisors, for their unwavering support and collaboration throughout this project. Your insights into the practical implications of integration testing have enriched our understanding and provided a real-world perspective on the subject.

We extend our heartfelt thanks to Ali Hajian Foroushany, Machine verification tester, Björn Nilsson, and Tamidala Sowmaya, Software Engineers, for their support during the summer vacation. Your willingness to share your knowledge and provide assistance has been invaluable, and we are grateful for the insights you offered during our interactions.

Lastly, We are indebted to our friends and family for their continuous encouragement, understanding, and belief in our capabilities. Your unwavering support has been a driving force behind every step we have taken on this academic journey.

In the following pages, we present the conclusion of our efforts, reflections, and findings. This report is a testament to the exploration and discoveries made in the realm of integration testing, and we hope it serves as a source of inspiration for further inquiries and contributions to this dynamic field.
# Contents

1 **Introduction**
   1.1 Background ........................................... 1
   1.2 Related Work .......................................... 2
   1.3 Problem formulation ................................... 3
   1.4 Motivation ............................................. 5
   1.5 Results ............................................... 6
   1.6 Scope/Limitation ...................................... 6
   1.7 Target group .......................................... 7
   1.8 Outline ............................................... 8

2 **Method**
   2.1 Research Project ...................................... 9
   2.2 Research methods ...................................... 11
   2.3 Reliability and Validity ............................... 12
      2.3.1 Reliability ....................................... 12
      2.3.2 Validity .......................................... 12
   2.4 Ethical considerations ................................ 13

3 **Theoretical Background** ............................... 15
   3.1 Software Testing ...................................... 15
   3.2 Integration Testing at ECU Level ..................... 16
   3.3 Simulation-based Testing .............................. 16
   3.4 Hardware-in-the-Loop Testing ......................... 17
   3.5 Black Box Testing Techniques ......................... 17
      3.5.1 Boundary Value Analysis ......................... 17
      3.5.2 Equivalence Partitioning ......................... 18

4 **Case Study** ........................................... 19
   4.1 Architecturally Significant Requirements ........... 19
   4.2 High-level System Design ............................. 21
      4.2.1 Functional Requirements ......................... 21
      4.2.2 Safety Requirements ............................. 24

5 **Research project – Implementation** ................. 26
   5.1 Testing Steps .......................................... 26
   5.2 Tools and Techniques Decision ....................... 28
      5.2.1 OpenModelica Software Simulator ............... 28
      5.2.2 Scenario-based Testing ........................... 29
         5.2.2.1 Normal Dumping Operation .................. 31
5.2.2 Load Dumping ................................................. 33
5.2.2.3 Automatic Lowering ........................................ 36
5.2.2.4 Emergency Stop ........................................... 38
5.2.2.5 Unseated Driver ........................................... 40
5.2.2.6 Speed Limit Activation ................................. 42
5.2.2.7 User Interaction ............................................ 43
5.2.2.8 Sensor Malfunction ........................................ 46
5.2.3 Boundary and Stress Testing ......................... 47
5.2.3.1 Boundary Value Analysis ............................... 48
5.2.3.2 Stress Testing ............................................. 49
5.2.4 Integration Testing ........................................... 50
5.2.4.1 Integration Testing - Functional Requirements .... 50
5.2.4.2 Integration Testing - Safety Requirements .......... 51
5.2.5 User Interaction Testing ................................. 52
5.3 Data Collection Overview ................................. 53
5.3.1 Data Collection ............................................. 53
5.3.2 Data Analysis ............................................... 54

6 Results ................................................................. 55
6.1 RQ1 ............................................................. 55
6.2 RQ2 ............................................................. 55
6.3 RQ3 ............................................................. 56

7 Analysis ................................................................. 57

8 Discussion ............................................................... 58

9 Conclusions .............................................................. 60

References ................................................................. 61
1 Introduction

Software engineering (SE) is a dedicated field containing software design, development, testing, and deployment. It involves transforming initial ideas into user-friendly programs that adapt to evolving customer requirements [1]. Embedded software engineering focuses on software development within various products such as vehicles, airplanes, and medical systems [2]. This domain emphasizes crucial factors, including reliability, cost-efficiency, and time-to-market, necessitating specialized development technologies tailored to meet these unique demands [2].

Software testing ensures program quality by executing controlled conditions to identify errors and validate conformance to specifications and user requirements [3]. Integration testing is a significant aspect of software testing that evaluates the interaction between modules to create a functional program [4]. It shifts the focus from individual statements to module interfaces [4]. The goal is to detect errors resulting from issues in component interaction encountered during the integration process, as these problems constitute a significant proportion of software errors [4].

This 15 HEC (Higher Education Credits) computer science bachelor thesis delves into the integrated software testing and verification domain within the embedded systems context, specifically applied to heavy machinery such as Volvo articulated haulers. The scope of the research concentrates on the dumping functionality within Volvo articulated haulers, enabling the provision of targeted insights and recommendations tailored to this specific and significant area of interest.

1.1 Background

"The rapid increase in the software complexity of today’s Electronic Control Units (ECUs) makes testing a central and significant task within automotive control software development" [5]. Defects in embedded systems, like ECUs, if left undetected, can pose significant risks to production and human safety [6]. As a result, about 50% of the total time in automotive development is dedicated to software testing [7].

ISO 26262 is an industry-specific standard focused on functional safety for automotive electrical and electronic systems, including ECUs [8]. It provides a comprehensive framework to minimize risks and hazards associated with machine software failures [8]. The ISO 26262 V-model aligns safety-related activities with different software development phases, identifying safety issues early and reducing the likelihood of defects [9]. This standard emphasizes testing and verification, including unit testing, integration testing, and requirements verification.

This research project focuses on Requirement-based and Scenario-based testing at the integration level for the dumping functionality. Integration testing involves multiple components like ECUs, sensors, and actuators to ensure seamless operation and adherence
to functional and safety requirements.

Testing ECUs and controlling modules in the Software and Hardware In The Loop (SIL/HIL) environment presents a notable challenge. Specifically, achieving effective testing and verification of integrated software systems is complex, aiming to ensure functional accuracy and adherence to safety standards. The intricate interplay of diverse components—ECUs, sensors, actuators, and communication protocols—in a dynamic environment necessitates robust testing strategies. Moreover, strict compliance with standards such as ISO 26262 amplifies the intricacy, demanding meticulous consideration of safety throughout the development life cycle.

1.2 Related Work

In challenges exploration related to integration testing within HIL test rig environments, various studies have delved into analogous issues within the domain of embedded systems.

Schroeder et al. delved into the challenges faced by integration and test engineers at a prominent automotive original equipment manufacturer (OEM) during software integration testing in interconnected HIL test rig environments [10]. This paper presents findings obtained through interviews with engineers, shedding light on the challenges, underlying causes, and potential solutions tied to HIL test rigs [10]. Challenges highlighted encompassed handling ambiguous requirements and unclear semantics within system interfaces, inconsistencies in type information among third-party software, issues stemming from faulty and outdated third-party software, and the necessity for specialized expertise for specific software components [10]. The study emphasizes that integration testing in HIL test rig environments is a resource-intensive and complex task that often involves collaboration among multiple parties from different suppliers [10].

Xu et al. presented a significant development and testing aspect of control execution and decision algorithms for autonomous systems. Existing methodologies often involve using tools like Matlab/Simulink for control execution. The simulation platform enables quick functional verification in the early stages of development, thereby improving development efficiency [11]. However, the separate development stages of these algorithms can pose challenges when conducting Software-in-the-Loop (SIL) tests [11]. Considering these challenges, research proposes innovative SIL simulation platforms that bridge the gap between various software components [11]. The methodology used in the paper revolves around establishing a simulation platform that facilitates seamless model integration, message compilation, topic definition, and communication connection [11]. While the domain of heavy machinery introduces unique challenges, such as integration testing within the context of articulated hauler dumping systems, the lessons learned from this related work underscore the significance of simulation-based platforms for algorithm validation and their potential to accelerate development cycles. Drawing upon these paral-
lels, this research endeavors to adapt and extend simulation-based testing methodologies to address the challenges raised by integration testing in heavy machinery applications.

1.3 Problem formulation

The domain of integrated software testing and verification within embedded systems presents a significant challenge, particularly in the context of heavy machinery such as Volvo articulated haulers. Existing related work [10] [11] underscores the complexities in integration testing in intricate systems characterized by numerous interactions. The ECU-level testing phase, coupled with simulator tools, emerges as a promising avenue for improving the testing and verification process. This phase allows integration testing on smaller subsystems with fewer interactions, offering the potential to streamline the process.

Figure 1.1: Automotive Software Testing V-model according to ISO 26262

A noticeable gap persists in the integrated software testing and verification, especially concerning heavy machinery applications. The lack of comprehensive methodologies and practices tailored to integration testing at the ECU level for heavy machinery software
creates a research void. Volvo CE’s testing and verification process, as illustrated in Figure 1.1, involves several integration testing phases. Among these phases, those about ECU integration are of particular interest. While Volvo CE currently performs integration testing of ECUs during the HIL simulator phase, this approach is resource-intensive and time-consuming.

This research addresses this challenge by proposing an alternative approach to conducting similar integration testing during an earlier stage using compact integrated ECUs. The primary aim is to uncover and rectify gaps related to ECU-level integration testing to improve software quality and reliability. Moreover, this approach could lead to a reduction in lead time, especially within safety-critical systems.

**Research Questions:**

1. **What are the suitable methods for conducting integration testing at the ECU level?**
   
   By exploring and evaluating various methodologies, our report can provide a comprehensive understanding of the testing strategies that align with the specific complexities of ECU integration in Volvo articulated haulers. The insights gained from this investigation can serve as a foundation for proposing innovative approaches that ensure seamless integration and optimal performance of ECUs, contributing to a robust testing framework development.

2. **Can a single testing method provide sufficient support for fault detection and coverage at the ECU integration level?**
   
   This question focuses on the effectiveness of a single testing method in identifying faults and achieving comprehensive coverage within the ECU integration level. The results of this inquiry can provide valuable insights into whether a unified testing approach can adequately address the diverse scenarios and interactions within heavy machinery systems. The findings will inform the selection of testing methods that offer maximum fault detection and coverage, ultimately leading to enhanced software quality, reduced vulnerabilities, and improved reliability in the integrated ECUs.

3. **How effective is ECU integration level testing in reducing feedback loop?**
   
   This research question addresses a critical aspect of development efficiency – the feedback loop. By examining how ECU integration level testing impacts the feedback loop, this report can quantify the improvements in development speed, agility, and responsiveness that arise from early-stage integration testing. This insight is invaluable for Volvo CE and other stakeholders. It offers a concrete measure of the benefits of implementing the proposed approach. The reduction in feedback loop time underscores the potential for quicker iterations, fewer delays, and enhanced
collaboration among development teams, ultimately leading to improved software quality and reduced time-to-market.

By exploring these questions, the study aims to contribute valuable insights and methodologies benefiting the articulated industry and other domains relying on embedded software for required functions.

1.4 Motivation

From a scientific standpoint, this research project on integrated software testing and verification in embedded systems significantly advances the software engineering field. As the complexity of ECUs in autonomous vehicle increase, the need for robust testing methodologies becomes paramount. The proposed integration testing framework, aligned with the ISO 26262 v-model, offers a systematic approach to validate functional requirements, promote safety, and minimize software defects in complex embedded systems.

Looking at it from a societal perspective, reliable software in embedded systems is of utmost importance, directly impacting human safety and production efficiency. The potential risks associated with vehicle software failures can lead to accidents, injuries, and even loss of lives. By emphasizing comprehensive testing and verification, this research project aims to enhance the safety and performance of Volvo CE, thereby contributing to safer load transportation. As the heavy equipment industry continues to evolve and integrate advanced software technologies into vehicles, ensuring the software’s functional safety and reliability becomes a critical societal well-being aspect.

From an industrial perspective, the findings and recommendations of this research hold significant value for manufacturers and software providers, particularly those involved in complex embedded systems development. The ISO 26262 standard serves as a benchmark for achieving functional safety, and by adhering to its guidelines, companies can enhance the quality and safety of their products. Implementing this framework can identify software defects earlier, improve software quality, and streamline development processes. By addressing the challenges posed by testing ECUs and controlling modules in the ECU-simulated environment, this study can help heavy equipment manufacturers deliver safer and more reliable vehicles to the market.

We should remain realistic in our claims, recognizing that the proposed integration testing framework and its alignment with the ISO 26262 V-model represent steps toward improving software quality and safety. While this research project aims to make valuable contributions to the field, it is crucial to acknowledge the limitations and challenges associated with the embedded software development domain. By combining scientific rigor, societal relevance, and industrial applicability, this research project strives to make a meaningful impact on the advancement of software testing practices and the safety of complex embedded systems.
1.5 Results

In the dynamic heavy machinery domain, ensuring successful integration and optimal operation of complex systems necessitates meticulous testing and validation protocols. This research project tackles the challenges inherent in integrating the articulated hauler dumping system—the complex interplay of components and sensors. Through a strategic amalgamation of advanced testing methodologies and simulation-based tools, this study aims to emphasize the effectiveness of the proposed testing approach in ensuring the system’s dependability, safety, and overall performance. However, in this thesis, we considered articulated hauler dumping scenarios to show our results, but our work can benefit other use cases like acceleration and loading and other domains like aerospace, railway, and health.

The implementation and testing of the articulated hauler dumping system have yielded valuable insights into its performance and alignment with the specified Architecturally Significant Requirements (ASRs). A comprehensive testing approach, including boundary testing, stress testing, integration testing, and user interaction testing, was instrumental in evaluating the system’s behavior and capabilities. The validation mechanism involved feedback from domain experts and stakeholders. Relevant experts in the articulated hauler systems field, embedded software, and testing practices reviewed the results, methodologies, and analysis. Their valuable input played a pivotal role in identifying potential errors, gaps, or inconsistencies in the findings, significantly enhancing the accuracy and credibility of the results.

1.6 Scope/Limitation

The domain of Embedded Software Development is expansive and diverse. This study centers exclusively on software development for Heavy Machinery. Further, the research narrows the focus to Volvo articulated haulers, complex ecosystems with numerous ECUs and functionalities. Consequently, the study delves into testing and verifying the dumping functionality within the Volvo hauler system.

In the ISO 26262 automotive software development v-model, testing and verification are categorized into three phases: unit testing, integration testing, and safety requirements verification. Among the three distinct testing phases where integration testing can be applied, as depicted in Figure 1.1, this research concentrates on integration testing during the ECU phase. It involves exploring requirements that span multiple ECUs across various chassis parts and creating potential scenario-based test cases. Notably, the analysis focuses solely on high-level requirements within the Software Development Life Cycle.
1.7 Target group

The target audience for this research project includes professionals and stakeholders involved in embedded software systems development and testing. Specifically, the following target groups can benefit from the findings and recommendations of this study:

- **Embedded Software Developers and Engineers**: Professionals working in the embedded software development domain, especially those responsible for designing and implementing ECUs and safety-critical functionalities, will find valuable insights in this research. Embedded software developers can utilize this framework to enhance their testing practices, leading to more reliable and safer software components.

- **Safety and Compliance Experts**: Individuals responsible for ensuring compliance with industry standards and safety regulations, such as the ISO 26262 standard, will find relevance in this research. The study explores the application of the ISO 26262 V-model in the context of integrated software testing for Volvo articulated haulers. Safety and compliance experts can use the insights gained from this research to evaluate and improve their safety assessment processes and verification activities.

- **Autonomous Manufacturers and OEMs**: OEMs and companies involved in articulated haulers production, including heavy machinery, are a crucial target group for this research. The proposed integration testing framework can assist manufacturers in enhancing overall quality and safety. By adopting the ISO 26262 V-model and aligning testing practices with stakeholder requirements, manufacturers can deliver more reliable and safer vehicles to the market.

- **Academic and Research Community**: The research project contributes to the academic and research community in the software engineering field, particularly those interested in embedded systems and software testing. Researchers and scholars studying integrated software testing methodologies and safety-critical systems can draw upon the findings of this study for reference and further exploration.

- **Software Testing and Verification Experts**: Professionals specializing in software testing and verification will find relevance in the proposed integration testing framework and its alignment with the ISO 26262 V-model. The study explores the challenges and considerations in testing ECUs and controlling modules in the SIL environment. Software testing experts can gain insights into addressing complexities associated with testing embedded systems.
1.8 Outline

This paper includes several chapters, each contributing to a comprehensive understanding of the research. Chapter 1 encompasses the introduction, presenting the background, related work, scope and limitations, motivation, and preliminary results. In Chapter 2, we delve into the methodological framework and research methods utilized in this study. Building on this, Chapter 3 delves into the theoretical background, defining key concepts, identifying the knowledge gap, and highlighting pertinent challenges. Chapter 4 details the specifics of case study implementation and related activities. Chapter 5 showcases the outcomes of these studies in the form of results. Subsequently, Chapter 6 undertakes an in-depth analysis of these results, shedding light on their implications. In Chapter 7, the completion of previous chapters converges as it discusses the conclusions drawn from this study. Additionally, it outlines potential avenues for future research endeavors, guiding the trajectory of subsequent investigations.
2 Method

In this section, the research methodology employed in this study is discussed, showcasing a comprehensive understanding of the research methods utilized to address the identified knowledge gap and answer the research questions. The chosen research method for this study is a combination of Case Study, Design Science, and Literature Review. These methods align with the research objectives and provide a structured approach to exploring integration testing practices within the specified domain.

2.1 Research Project

This study investigates integration testing practices specifically at the ECU phase and their alignment with stakeholders’ functional and safety requirements within the context of Volvo CE. The project strategically follows the principles of the Design Science methodology, which involves a systematic approach to defining, solving, and validating real-world problems through innovative artifact creation. By adopting this methodology, the research aims to develop a reliable integration testing framework for ECU-level testing that harmonizes with functional requirements and enriches the software development process at Volvo CE.

In [12], the choice of the Design Science methodology underpins its inherent capacity to systematically tackle real-world problems by constructing practical solutions. This approach is particularly apt for this research project, as it aligns with the overarching goal of crafting an effective integration testing framework that caters to the unique demands of Volvo CE’s software development environment. Adhering to the Design Science methodology, the study will address the research questions and generate tangible contributions for academia and industry.

Project Roadmap

1. Comprehend stakeholders’ and supervisors’ needs:
   - Activity: Collect functional and safety requirements from stakeholders relevant to integrated scenarios.
   - Activity: Engage with supervisors to obtain insights into the system architecture requirements.
   - Objective: Understand the nuances of stakeholder and supervisor needs to report the framework design.

2. Problem Identification and Objective Setting:
   - Activity: Identify existing challenges and gaps in the current software development and testing environment at Volvo CE.
• Activity: Evaluate the potential benefits of integrating testing at the ECU level to enhance software quality and functional validation.

• Objective: Clearly define the problem space and establish purposes for the proposed framework.

3. Literature Review:

• Activity: Conduct a literature review to comprehend best practices and existing approaches to integration testing in embedded systems development.

• Activity: Collect data from relevant literature, industry reports, and documentation pertinent to integration testing practices.

• Objective: Create a comprehensive understanding of the current landscape to inform the framework’s design and validation.

4. Case Study:

• Activity: Perform a meticulous case study within Volvo CE’s enterprise architect system model to collect pertinent data.

• Activity: Develop a plan for generating comprehensive test cases based on the requirements gathered.

• Objective: Gather empirical data to refine the framework design and validate its applicability.

5. Testing Technique Evaluation:

• Activity: Explore and evaluate diverse integration testing techniques suitable for ECU-level testing in Volvo CE’s environment.

• Objective: Determine the most suitable testing technique for stakeholder requirements.

6. Mock Testing and Coverage Analysis:

• Activity: Execute mock testing using developed test cases to evaluate functional and safety requirements coverage.

• Activity: Analyze achieved test coverage, identifying strengths and weaknesses.

• Objective: Assess the comprehensiveness of the testing technique and its ability to meet requirements.

7. Recommendations and Framework Proposal:

• Activity: Propose a comprehensive integration testing solution tailored to Volvo CE’s enterprise architect system model.
Objective: Formulate a practical solution and recommendations grounded in empirical findings.

This meticulously crafted project roadmap harnesses the principles of the Design Science methodology to holistically address the identified challenges and enhance software testing practices within Volvo CE’s domain.

2.2 Research methods

In a comprehensive investigation of the effectiveness of integration testing at the ECU level, we have carefully selected a range of research methods that align with the specific research objectives, stakeholders’ interests, and the contextual setup of our thesis project.

1. Literature Review:

   • Description: The research begins with an in-depth literature review delving into established practices and state-of-the-art approaches to integration testing in embedded systems domain development and related industries.

   • Motivation: The literature review serves as the foundation of our research, equipping us with a comprehensive understanding of contemporary integration testing methodologies, industry standards, and the latest research advancements. By comparing these findings with Volvo’s existing practices, we ensure that our proposed integration testing framework addresses the current challenges and aligns with industry trends and best practices.

2. Case Study:

   • Description: A meticulous case study will be conducted within Volvo’s CE enterprise architect system model to gather primary data encompassing functional and safety requirements and insights into the existing software development and testing process.

   • Motivation: This research method provides a valuable opportunity to immerse ourselves in the intricacies of constructing test cases, navigating the complexities of multiple ECU integrations, and comprehending the regulatory landscape. The case study empowers us to gather firsthand data on stakeholders’ requirements, allowing for a deeper understanding of their expectations. While our original intention was to test the solution within Volvo’s testing environment, time limitations, and logistical complexities led us to opt for this alternative approach.

By implementing this diverse range of research methods, we aim to achieve a holistic understanding of the challenges and opportunities in integration testing at the ECU level in Volvo’s testing environment.
2.3 Reliability and Validity

In any research project, ensuring the reliability and validity of the results is paramount to maintaining the integrity of the study. When conducting the research multiple times using the same methods, reliability refers to the consistency and repeatability of the results. Validity concerns the accuracy and truthfulness of the conclusions drawn from the data collected.

2.3.1 Reliability

To uphold the research reliability, we have taken several steps:

1. Research Design Consistency: A consistent research design, based on the design science approach, has been employed throughout the study. This standardized approach facilitates the systematic development and implementation of the integration testing framework, reducing the chances of biased outcomes.

2. Data Collection Methods: We precisely chose the methods to minimize potential biases and inconsistencies. The SysML requirements diagrams for gathering stakeholders’ functional requirements ensure standardized data representation and increase the reliability of the collected information.

3. Reproducibility of Testing: We documented the testing process, including mock testing and coverage analysis, to enable other researchers to reproduce the experiments and verify the results independently.

4. Clear Documentation: Detailed documentation of all steps involved in the case study and mock testing ensure transparency and allow for verification by others.

5. Multiple Data Sources: To strengthen reliability, we gather data from multiple sources, including literature reviews, industry reports, and stakeholders’ requirements. Triangulating data from diverse sources enhances the robustness of the research outcomes.

2.3.2 Validity

Validity refers to the accuracy and appropriateness of research findings, ensuring that the results effectively address the research questions and objectives. We have implemented the following measures to maintain the validity of this research:

1. Research Scope: We have carefully formulated the research questions and objectives to address the specific scope of this study, which is integration testing at the ECU level. This focused approach ensures that the findings directly relate to the problem formulation.
2. Case Study Design: We ensured a meticulous plan and execution of the case study within the Volvo CE enterprise architect system model to capture stakeholders’ functional and safety requirements. This approach ensures that the integration testing framework precisely aligns with the actual needs of the Volvo CE development process.

3. Data Analysis: The data collected from the case study and other sources will be analyzed using established research methods. The chosen data analysis techniques will be appropriate for addressing the research questions and validating the proposed integration testing framework.

4. Software Engineers Engagement: We actively sought collaboration with Volvo supervisors throughout the research process. Involving Software Engineers ensures that the research findings are relevant and applicable to real-world scenarios, enhancing the external study validity.

5. Limitations Acknowledgment: The limitations of the research, such as the time constraints and complexities involved in testing within the Volvo CE testing environment, have been acknowledged. Addressing these limitations helps to maintain the internal study validity and ensures the accuracy of interpretations.

By accepting these reliability and validity measures, this research provides trustworthy and meaningful insights into integration testing at the ECU level and its potential contributions to the embedded software development domain.

2.4 Ethical considerations

In conducting this research project, we meticulously addressed ethical considerations to protect participants’ rights, privacy, and the overall integrity of the research process.

- Confidentiality and Data Privacy: Data collected during the case study, including System Architecture, Volvo Bitbucket, and Confluence pages, are considered sensitive and confidential. Using publicly available or research-approved data guarantees confidentiality and privacy by exclusively concentrating on data that does not risk compromising sensitive or proprietary information.

- Sampling and Bias: The case study conducted within the Volvo CE enterprise architect system model involves data collection from various stakeholders, including supervisors and other personnel involved in the development and testing process. To minimize bias, a diverse and representative sample of stakeholders will be targeted to ensure a comprehensive understanding of requirements and challenges.
• Risk of Harm: Given the nature of the research, the risk of harm to participants is minimal, as the study does not involve any human subjects directly. The research focuses on software development and integration testing in the context of Volvo construction equipment, which does not pose any physical or psychological risks to individuals.

• Participation and Consent: Participants in the study, specifically Volvo supervisors, are fully aware of their involvement and have provided informed consent to participate. Before data collection, we notify participants about the research objectives, data collection methods, and the potential uses of the collected information. We emphasize voluntary participation, and participants are assured the freedom to withdraw from the study at any point without facing consequences.

• Ethical Approval: This research project went through the review process. The appropriate institutional review board assessed ethical considerations during this process. Any potential ethical concerns were addressed and mitigated following research ethics guidelines.

Throughout this study, the research team upheld the highest levels of integrity and adhered to ethical principles and guidelines. Our commitment includes maintaining confidentiality, protecting data privacy, securing informed consent, and mitigating potential biases. By doing so, we aim to generate reliable and valid findings while respecting the rights and privacy of all participants involved.
3 Theoretical Background

This chapter explores the fundamental concepts that form the basis of the subject matter in this study. It provides a comprehensive theory exploration, principles, and established frameworks relevant to the research topic. Exploring the theoretical foundations in this section shows a robust knowledge foundation, enabling readers to grasp the study context, significance, and broader implications.

3.1 Software Testing

Software testing plays a vital role in software development by aiming to detect errors within the software development life-cycle. It involves executing the software to find any inconsistencies between the expected and actual code behavior to verify the quality and functionality of the software. Some software testing techniques include unit testing, which tests individual components, and integration testing, which verifies the interaction between different sub-systems.

Test cases are critical in verifying software behavior and functionality, often involving multiple steps for comprehensive testing and validation. Various techniques, including Requirement-based Testing, Model-based Testing, and Code-based Testing, can be used to approach test case creation. Requirement-based testing involves creating test cases based on the specified inputs, outputs, and functionality outlined in the requirements [13]. In Model-based testing, test cases are derived directly from a system model under development, which provides benefits such as reduced errors and increased efficiency, although it may require expertise in model syntaxes and semantics [14]. In Code-based Testing, testers execute test cases to ensure comprehensive coverage of system test paths [15]. This technique allows for early error detection and cost reduction as it is performed at the beginning of development when most bugs originate from requirements [15].

Additionally, Other testing methodologies, such as Stress Testing, Scenario-based Testing, and User Interaction Testing, have been introduced to ensure comprehensive understanding. Stress Testing evaluates system robustness under extreme conditions, assessing its ability to endure strain beyond regular operational scenarios [16]. Scenario-based Testing constructs test cases to simulate real-world scenarios, exposing potential defects in specific operational contexts [17]. User Interaction Testing, on the other hand, gauges how the system responds to user inputs and commands [18]. Using these methodologies contributes to a more holistic approach to software testing, enhancing the identification of defects across diverse usage scenarios.

It is important to note that software testing alone cannot guarantee the correctness of a program. It only highlights the presence of defects. Early defect detection is crucial in keeping costs down, as the cost of fixing defects increases exponentially throughout the development cycle. Detecting defects during testing can be 15 times more expensive
than identifying them during system design and 100 times more if they go unnoticed until system maintenance [19]. This cost factor has driven the adoption of early testing and testing integration during the development cycle.

### 3.2 Integration Testing at ECU Level

Integration testing, a critical phase in the software development lifecycle, systematically integrates software components to assess their interactions and functionalities. Depending on the system’s complexity and requirements, this testing phase can occur in various environments, such as software simulators or HIL setups. These environments offer controlled spaces to validate the behavior and performance of the integrated components. Modern vehicles, including Volvo articulated haulers, utilize ECUs connected through a communication protocol known as the Controller Area Network (CAN bus).

CAN bus is a communication protocol used in vehicles to allow different ECUs to communicate with each other efficiently and reliably [20]. CAN network facilitates the exchange of data and information between these ECUs, allowing for the sharing of sensor data, calculated input, and control signals. By distributing functions across multiple ECUs and connecting them via CAN, vehicles can achieve greater efficiency, flexibility, and scalability in their operations [20]. This architecture enables different ECUs to work together and share relevant data.

The primary purpose of ECUs is to manage and control various functions within the machine. As the number of parts and complexity grows, the number of ECUs also increases. ECUs are responsible for performing specific tasks in the vehicle. These ECUs receive inputs from sensors and other ECUs, process the information, and generate outputs for actuators, including components like engines, brakes, and the dumping mechanism. The number of ECUs and their complexity increases when the system requires new functionalities to manage an expanding range of functions.

### 3.3 Simulation-based Testing

Simulation-based testing is a testing approach that utilizes software simulations to create virtual environments mocking real-world scenarios and conditions [21]. This approach is employed to validate software functionalities, test system behaviors, and assess the performance of software components in a controlled and virtual setting [21]. Software simulation testing does not require the actual physical hardware components’ use; instead, it relies on software models to replicate the responses and interactions in real-world scenarios [21].

Software simulation testing is an invaluable method for evaluating the behavior of ECUs and interactions within the articulated hauler dumping system. This approach involves creating software models that mimic the ECUs’ responses, the Dump Lever’s
puts, sensor measurements, actuator outputs, and the communication signals exchanged between these components. It is practical in early-stage testing, scenario testing, and rapid iterations to fine-tune software behaviors. Additionally, this approach provides a cost-effective way to identify potential issues, evaluate system responses, and refine software functionalities before moving to more resource-intensive and hardware-dependent testing stages.

3.4 Hardware-in-the-Loop Testing

HIL testing refers to a specialized testing methodology used to validate and verify complex systems, particularly those involving ECUs and their interactions with real hardware components [22]. HIL testing creates a controlled testing environment where actual hardware components integrate with simulation models representing the remaining components of the system [22].

In this setup, the hardware components interact with the simulation models in a closed-loop manner, replicating real-world conditions and interactions [22]. HIL testing assesses performance, functionality, and system interactions under various scenarios without the entire physical system being fully assembled and operational.

HIL testing identifies potential issues, such as software bugs, timing discrepancies, and hardware-software integration challenges before deployment. It offers a realistic and efficient way to validate system behavior, improve system reliability, and reduce the risk of failures in operational environments.

3.5 Black Box Testing Techniques

Black box testing is a software testing technique that concentrates on a system’s overall functionality without requiring any insight into its internal mechanisms [23]. This approach builds upon the requirements provided by customers, focusing on the perspective of end users. The primary objective of black box testing is to validate that the system’s functionality aligns with specified requirements and uncover any discrepancies or unexpected behavior. During black box testing, testers systematically examine valid and invalid inputs to ensure that the system responds as expected to a range of scenarios. This methodology aids in the identification of any flaws or errors in the system’s handling of incorrect or unexpected inputs.

3.5.1 Boundary Value Analysis

Boundary Value Analysis (BVA) is a black-box testing technique employed in software testing to identify potential defects or errors at the boundaries or edges of valid input ranges [24]. This method ensures that the software behaves correctly at the extremities of
its acceptable input values [24]. By selecting input values just at, below, or above these boundaries, testers can uncover issues that might arise due to specific conditions [24].

3.5.2 Equivalence Partitioning

Equivalence Partitioning is an approach used in software testing to categorize input data into groups or partitions that behave similarly [24]. This technique streamlines the testing process by reducing the number of test cases while retaining effective coverage [24]. Boundary Value Analysis and Equivalence Partitioning, group inputs into classes that exhibit similar behavior [24]. These techniques help testers create more focused and effective test cases, improving the coverage of different scenarios and increasing the likelihood of detecting defects [24].
4 Case Study

In this chapter, our case study illustrates the practical application of the concepts and methodologies discussed earlier. The case study centers around an articulated hauler dumping system, a complex and integral component of modern heavy machinery. This system encapsulates a network of interconnected elements, including Electronic Control Units (ECUs), sensors, actuators, and user interfaces, all working to facilitate precise and controlled dumping operations.

The case study offers insights into the stakeholders’ considerations when designing, implementing, and testing such intricate systems. By examining the real-world scenario of the articulated hauler dumping system, we gain a deeper understanding of how architectural decisions, integration strategies, and testing approaches play out in practice.

Throughout the case study, we navigate various stages of the system’s life cycle, starting with requirements analysis and architectural design. By dissecting each phase, we uncover the strategies to address architecturally significant requirements, ensure seamless integration among components, and validate the system’s behavior under diverse scenarios.

4.1 Architecturally Significant Requirements

ASRs are significant requirements that impact the software system architecture [25]. Considering these requirements during the design and development of a software system is crucial, given their potential impact on the system’s structure and behavior [25].

In the context of an articulated hauler dumping system, several ASRs could include:

- **Real-time Responsiveness**: The system must respond to user inputs and sensor data in real-time to ensure safe and accurate control of the dumping process. This requirement might influence the choice of hardware components, communication protocols, and software architecture to minimize latency.

- **Reliability and Safety**: The system must operate reliably and safely to prevent accidents or malfunctions. This ASR could affect the redundancy of critical components, fault tolerance mechanisms, and the overall system architecture to ensure fail-safe operations.

- **Interoperability with Existing Systems**: The articulated hauler dumping system needs to interact with other existing systems or equipment to support seamless integration. This requirement could lead to decisions on communication protocols, data formats, and interfaces.

- **User-Friendly Interface**: The user interface of the Machine Interface should be intuitive and user-friendly, allowing operators to interact easily with the system. This
requirement could influence the design of the graphical user interface (GUI) and the overall user experience.

- Data Logging and Analytics: The system might need to log data for analysis and diagnostics. This requirement influences architectural decisions for data storage, retrieval, and analytics capabilities.

- Maintainability and Upgradability: The system should facilitate easy maintenance and upgrades. This requirement might impact decisions on software architecture, component encapsulation, and version control practices.

- Integration with External Sensors and Actuators: The system should interact with external sensors and actuators beyond the provided components for seamless integration. This ASR could involve decisions on communication protocols and interface standardization.

- Modularity and Scalability: The architecture should allow for the modular expansion of new functionalities or components, ensuring that the system can accommodate enhancements without disrupting existing functionalities.

- Power Efficiency: The system should optimize power consumption during different operational scenarios, identifying areas where power usage optimization does not compromise system performance.

- Security and Authentication: The architecture should incorporate security measures to identify potential vulnerabilities and ensure proper authentication mechanisms.

- Regulatory Compliance: The system should adhere to industry-specific regulations and standards. Architectural decisions should ensure that the system meets the required compliance criteria.

Identifying and addressing these architecturally significant requirements play a crucial role in shaping the testing approach of a software system. They can significantly influence the testing strategy, methodologies, and priorities.
4.2 High-level System Design

This section presents an overview of the high-level design of the dumping sub-system, encompassing a range of components, interactions, and functionalities that collectively facilitate articulated hauler dumping operations. At its essence, the dumping system comprises a network of interlinked sub-systems, including the Dump Lever, Seat Sensor, ECU1, ECU2, ECU3, ECU4, Dump Body Angle Sensor, and the Machine Interface. These integral elements collaborate harmoniously to realize diverse functionalities and validate safety, each playing a vital role in orchestrating the seamless and precise execution of the dumping process.

4.2.1 Functional Requirements

![Diagram of high-level system design]

Figure 4.2: High-Level Design of Dumping Functional Requirements

Figure 4.2 shows the description of the system design based on the functional requirements:

- **Dump Lever:**
  - The Dump Lever provides an analog input, signifying the intent to raise the dump body.

- **Rising Dump Body:**
  - Dump Lever sends an analog input to ECU1.
  - ECU1 is responsible for different inputs from the Dump Lever and sends it as a CAN signal to ECU2, indicating the raise, lower, or hold action.
- ECU2 further communicates with ECU3, issuing a PWM\(^1\) request to initiate the body elevation.
- ECU3 translates the PWM signals into appropriate control signals for the dump valves, facilitating the body’s upward movement.

**Lowering Dump Body:**

- Dump Lever sends an analog input to ECU1.
- ECU1 communicates the desired action to ECU2 via a CAN signal, specifying lowering.
- ECU2 transmits a PWM request to ECU3, prompting the body to descend.
- ECU3 translates the PWM signals into control commands to lower the dump body.

**Holding Dump Body:**

- The Dump Lever’s analog input informs ECU1 about the intention to hold the body.
- ECU1 conveys this status to ECU2 via a CAN signal.
- ECU2, recognizing the hold command, deactivates PWM requests to ECU3, effectively maintaining the body’s current position and transmits the dump body angle to the Machine Interface for the user.
- ECU3 simultaneously stops the dump valves.

**Automatic Lowering:**

- When the Dump Lever provides an analog input for automatic lowering, ECU1 acknowledges this through a CAN signal.
- ECU1 communicates with ECU2, initiating a request for automatic body descent.
- ECU2 engages ECU3, which issues PWM signals to the dump valves, orchestrating the controlled lowering of the body.

---

\(^1\)PWM stands for Pulse Width Modulation. It is a technique used to control the amount of power delivered to a device by varying the width of the pulses in a digital signal [26]. PWM finds application in various domains, including motor control, hydraulic valves, LED dimming, and audio signal generation. It enables precise control over a device output power or intensity by adjusting the duty cycle of the pulses [26]. PWM signals are typically generated by microcontrollers or specialized PWM controllers [26].
• Measure and Display Dump Body Position:
  – The Dump Body Angle Sensor continuously measures the dump body’s orientation angle.
  – This data goes to ECU2, which calculates the precise body position.
  – ECU2 communicates the calculated position to the Machine Interface, allowing real-time display for the users.

• Hard Stop Dampening:
  – When the Machine Interface issues a hard stop command, ECU2 receives this instruction.
  – ECU2 promptly terminates PWM requests to ECU3, halting any ongoing body movement.
  – Simultaneously, the Machine Interface is updated with the hard stop status for user notification.

In this holistic high-level design, the interconnections of the components, combined with their precise functionalities, result in a comprehensive system capable of autonomously managing the articulated hauler dumping process while ensuring user interaction. The ECUs, Dump Lever, Dump Body Angle Sensor, and Machine Interface orchestrated actions create a synergistic system that achieves controlled and efficient dumping operations.
4.2.2 Safety Requirements

Safety Requirements
1. Limit the machine speed if body is up.
2. Stop lowering and raising if speed limit exceed.
3. Hold dump body if driver is not seated.

Figure 4.3: High-Level Design of Dumping Safety Requirements

Figure 4.3 shows the description of the system design based on the safety requirements:
• Seat Sensor:
  – The Seat Sensor provides a digital input to show if the driver is seated on the seat.

• Limit the Machine Speed:
  – ECU2 shall send a CAN signal to ECU4, indicating the body is not in the frame and limiting the speed.
  – ECU4 further communicates with ECU3 to reduce the machine speed.

• Holding Dump Body when Driver not Seated:
  – The Seat Sensor digital input informs ECU1 about the driver stand-up.
  – ECU1 conveys this status to ECU2 via a CAN signal.
  – ECU2, recognizing the hold command, deactivates PWM requests to ECU3.
  – ECU3 simultaneously stops the dump valves.
5 Research project – Implementation

This chapter describes the activities undertaken to gather data for the case study. It reports all designs and implementations executed during the research project.

5.1 Testing Steps

This section describes our proposed solution that Volvo CE and similar manufacturers can implement (see Figure 5.4).

![Testing Steps Data Flow](Figure 5.4: Testing Steps Data Flow)

1. Software Simulator:
   - Virtual environment
   - Visual animations
   - Output reports
   - Integration with other systems

2. Scenario-based Testing
   - Various scenarios and conditions

3. Boundary and Stress Testing
   - Extremes of system capabilities
   - Stress scenarios

4. User Interaction Testing
   - UI mockups or simulations
   - User input and feedback testing

5. Integration Testing
   - ECU, sensors, actuators, and Machine Interface
   - Multi-component interactions

6. Peer-review and Expert Feedback
   - Collaboration for review
   - Feedback gathering
• Use a simulation platform to create a virtual environment replicating the articulated hauler dumping system.
• To ensure optimal comprehension and engagement, a well-executed visual animated display is crucial.
• The system’s utility enhances its ability to generate output reports and summary statistics that are comprehensive and interpretive.
• Seamlessly dovetailing into the organizational framework, the system should align with other relevant systems, promoting a suitable approach to operations.

2. Scenario-based Testing:
• Create a variety of scenarios that cover different operational conditions and user interactions.

3. Boundary and Stress Testing:
• Perform testing at the extremes of the system’s capabilities.
• Evaluate how the system behaves under stress, ensuring stability and responsiveness.

4. Integration Testing:
• Given the interconnecting of the system’s components, conduct integration testing to verify that different ECUs, sensors, actuators, and the Machine Interface communicate effectively and work together as intended.
• Focus on scenarios that involve multiple components interacting, in sequence or simultaneously.

5. User Interaction Testing:
• Develop user interface (UI) mock-ups or simulations to replicate how users interact with the Machine Interface.
• Validate that the system can correctly interpret user inputs and that the Machine Interface provides accurate feedback.

6. Peer Review and Expert Feedback:
• Collaborate with colleagues, mentors, or domain experts to review the testing approach and scenarios.
• Seek feedback to ensure test cases cover all relevant aspects and potential edge cases.
• Gathering Feedback can be through Slack, Microsoft Teams, and reviewing merge requests.
5.2 Tools and Techniques Decision

5.2.1 OpenModelica Software Simulator

OpenModelica\(^2\) is a compelling and advantageous option for simulating and testing articulated hauler dumping systems, thanks to its unique combination of features. There is a collection of free and commercial Modelica libraries suitable for simulating dumping functionality, including the Fluid Power Library, which is essential for modeling the hydraulic systems involved in the dumping mechanism [27]. Hydraulic components play a critical role in controlling the dumping functionality, ensuring smooth and controlled movements, and Flexible Bodies Library, which contains flexible modules to accurately represent the behavior of certain parts in the dumping damper, especially considering safety aspects related to material flexibility and stress [27].

Figure 5.5 illustrates a simplistic representation of an autonomous vehicle operating without a driver. OpenModelica, as depicted in the image, can simulate diverse vehicle and machinery components, accounting for varying weather conditions and road situations. This simulation versatility allows comprehensive analysis and evaluation of the system’s performance.

Figure 5.5: OpenModelica View

OpenModelica’s foundation in the equation-based object-oriented Modelica language allows the creation of highly detailed and accurate models of complex systems like articulated haulers. This approach captures intricate system behaviors, interactions, and dynamics with a remarkable level of fidelity. The equation-based nature of Modelica offers flexibility and abstraction, enabling the precise modeling of various components

\(^2\)https://openmodelica.org/
within the articulated hauler system [28]. This flexibility extends across different physical domains, encompassing mechanical modules, sensors, and control systems, all within a unified framework [28].

Moreover, OpenModelica efficiently handles large-scale simulations, a crucial capability for articulated hauler dumping systems that involve numerous interconnected components and interactions [28]. OpenModelica integration with other programming languages like C++ and Python further expands their capabilities, enhancing simulation setups and analyses by leveraging a broader range of tools and libraries.

OpenModelica’s open-source nature fosters a dynamic community of developers and users contributing to its ongoing development and improvement. This collaborative community ensures consistent updates, bug fixes, and enhancements, creating an evolving software environment that adapts to changing needs. Educational resources and user-friendly interfaces provided by OpenModelica help users overcome the learning curve associated with Modelica’s modeling and simulation approach.

Additionally, OpenModelica’s standardized approach to modeling and simulation facilitates easy collaboration among users and simplifies the exchange of models. The cost-efficient nature of an open-source tool eliminates licensing costs, making it a cost-effective option for research and development projects. OpenModelica’s interdisciplinary applicability is evident in its capacity to model multiple physical domains, aligning well with the multifaceted nature of articulated hauler systems.

OpenModelica enables the creation of reusable libraries, reducing model development time by allowing users to build upon existing models and components [28]. However, it’s essential to consider the performance limitations, such as restricted constant evaluation and expression simplification, which could impact simulation speed. Nonetheless, for integration testing purposes, where the focus is on assessing interactions and interoperability, these limitations might not be as critical as they would be for performance optimization.

OpenModelica offers robust equation-based modeling, scalability, integration capabilities, and an active open-source community. Its potential to accurately simulate complex interactions within articulated hauler dumping systems can have valuable testing, validation, and optimization insights. Despite the learning curve, OpenModelica’s benefits make it an invaluable project advancement tool.

5.2.2 Scenario-based Testing

Implementing scenario-based testing in the system involves creating a set of diverse and realistic scenarios that cover various operational conditions and user interactions. These scenarios serve as the foundation for testing each functionality of the articulated hauler dumping system. Here’s a step-by-step guide on how to implement scenario-based testing (see Figure 5.6):

29
1. Identify Scenarios: Begin by identifying a range of scenarios that the articulated hauler dumping system might encounter during its operation. Consider different use cases, environmental conditions, and user interactions the system needs to handle.

2. Define Test Cases: For each scenario, define specific test cases that outline the steps and conditions involved. Each test case should include a description, inputs, expected outcomes, and relevant preconditions.

3. Scenario Variations: Create variations that cover different possibilities within each scenario. For example, if you’re testing the "Rising Dump Body" functionality, you might have variations like "Rising Dump Body with Full Load" and "Rising Dump Body with Partial Load."

4. Input Generation: Prepare the necessary inputs for each test case. These inputs could involve providing analog input values for the Dump Lever, specifying the dump body angle for the Dump Body Angle Sensor, and simulating user interactions for the Machine Interface.

5. Execute Tests: Execute the test cases within the chosen simulation platform, such as OpenModelica. Make sure to provide inputs per the scenario and variation requirements. Monitor the system’s response to achieve expected outcomes.
6. Capture Data: During testing, capture relevant data, such as system behavior, responses, sensor readings, and any anomalies observed. This data will help analyze the system’s performance and identify areas for improvement.

7. Evaluate Results: Evaluate the results of each test case. Compare the actual outcomes with the expected outcomes. If there are discrepancies, investigate the root causes and determine if the system’s behavior aligns with its intended functionality.

8. Adapt and Refine: Based on the results of scenario-based testing, make necessary adjustments to the system’s components, algorithms, or configurations. Refine testing scenarios and test cases to cover a broader spectrum of possibilities.

9. Continuous Improvement: Incorporate the learned lessons from scenario-based testing into the system’s development process. Implement continuous improvement measures to enhance the system’s reliability, robustness, and performance.

10. Iterative Process: Scenario-based testing is an iterative process. After system growth or newly added functionality, revisit and expand testing scenarios to ensure comprehensive coverage of all possible scenarios.

5.2.2.1 Normal Dumping Operation

- **Scenario**: A standard dumping operation with a moderate load in the articulated hauler’s dump body.
Figure 5.8: Normal Dumping Operation rest of variation

- **Variation:**
  
  - Dumping on Different Terrain (see Figure 5.7): Simulate scenarios where raising the dump body on various types of terrain, such as level ground, uphill, downhill, or uneven surfaces. This variation evaluates the system’s ability to control the dump body’s movement on different terrains.

  - Dumping at Different Angles (Figure 5.8): Test the system’s response when dumping at different dump body angles. This variation ensures the system’s smooth transition between dump body positions and stability maintenance at various angles.

  - Dumping with Dynamic Load Changes (Figure 5.8): Simulate scenarios where the load in the dump body changes dynamically while the dumping operation is ongoing. This variation assesses the system’s adaptability to sudden load shifts.
– Dumping in Adverse Weather Conditions (Figure 5.8): Test the system’s performance in adverse weather conditions, such as rain, wind, or extreme temperatures. This variation evaluates the system’s robustness and reliability under challenging environmental factors.

– Dumping with Simultaneous Actions (Figure 5.8): Simulate scenarios after executing multiple functionalities, such as simultaneously raising and measuring the dump body position. This variation evaluates the system’s ability to manage concurrent actions effectively.

5.2.2.2 Load Dumping

• **Scenario:** Dumping operation with a heavy load in the articulated hauler’s dump body. (see Figure 5.9)

• **Variation:**
  
  – Maximum Load Capacity Dumping: Test the system’s performance when carrying and dumping loads at or near the maximum weight capacity of the articulated hauler’s dump body. This variation evaluates the system’s ability to manage the full load without compromising stability or control.

  – Uneven Load Distribution: Simulate scenarios after the heavy load uneven distribution within the articulated hauler’s dump body. This variation assesses the system’s capacity to handle imbalanced loads without tilting or malfunctioning.

  – Dynamic Load Shifting: Introduce scenarios where the heavy load shifts dynamically during dumping. This variation tests the system’s stability and adaptability to sudden changes in weight distribution.

  – Impact of Load Movement: Evaluate how the system responds when the heavy load shifts or moves due to the dumping action. This variation ensures that the system controls the load even as it shifts within the dump body.

  – Slow Dumping with Heavy Load: Test the system’s ability to raise and lower the heavy load at a slow, controlled pace. This variation handles system-increased forces and stresses associated with heavy loads.

  – Impact on Dumping Speed: Assess the system’s dumping speed and stability while carrying a heavy load. This variation evaluates how the system manages increased weight during the dumping process.

  – Heavy Load on Different Terrain: Simulate scenarios where the articulated hauler is carrying a heavy load on various types of terrain, such as uphill, downhill, or uneven surfaces. This variation assesses the system’s ability to control the heavy load in challenging conditions.
– Sequential Dumping with Heavy Load: Test the system’s performance when executing sequential dumping operations with a heavy load, including raising, partial lowering, and raising again. This variation evaluates the system’s accuracy and stability in complex sequences.

– Dumping with Load Compaction: Introduce scenarios where the heavy load compacts during the dumping. This variation tests the system’s ability to handle load compression and respond appropriately.

– Impact on Dump Body Angle: Evaluate how the system manages heavy loads at different dump body angles. This variation ensures that the system can handle the weight distribution effectively regardless of the dump body’s position.
**Scenario**: Dumping operation with a light load in the articulated hauler’s dump body. (see Figure 5.9)

**Variation**:

- **Minimum Load Capacity Dumping**: Test the system’s performance when dumping loads that are close to the minimum weight capacity of the articulated hauler’s dump body. This variation evaluates how accurately the system can control the dumping process with minimal load.

- **Precise Dumping Angle Control**: Simulate scenarios where the system is required to achieve precise dump body angles during light load dumping. This variation assesses the system’s maintainability, accuracy, and stability in controlling the dump body’s position.

- **Quick Dumping with Light Load**: Test the system’s responsiveness when raising and lowering the dump body with a light load rapid pace. This variation evaluates how well the system handles quick actions and adjustments.
– Dumping on Smooth Surfaces: Simulate scenarios where the articulated hauler operates on smooth-like paved surfaces. This variation assesses the system’s control and stability under less traction or resistance.

– Dumping with Load Shifting: Introduce scenarios where the light load shifts or moves within the dump body during the dumping process. This variation evaluates the system’s ability to manage load movements and maintain control.

– Impact on Dumping Speed: Assess the system’s dumping speed and stability while carrying a light load. This variation evaluates how the system manages the dumping process with minimal weight.

– Sequential Dumping with Light Load: Test the system’s performance during sequential dumping operations with a light load, including raising, lowering partially, and raising again. This variation evaluates the system’s accuracy and stability in complex sequences with light loads.

– Dumping on Inclined Surfaces: Simulate scenarios where the articulated hauler is on inclined surfaces while performing light load dumping. This variation assesses the system’s control and stability under different inclination angles.

– Dumping with Load Compaction: Introduce scenarios where the light load compacts during the dumping. This variation tests the system’s ability to handle load compression and respond appropriately to light loads.

5.2.2.3 Automatic Lowering

• Scenario: Initiating the automatic lowering feature with various load conditions. (see Figure 5.10)

• Variation:

  – Standard Automatic Lowering: Test the system’s response to the standard automatic lowering command, where the dump lever input triggers the automatic lowering process. This variation ensures that the system can execute the automatic lowering function reliably.

  – Automatic Lowering under Load: Simulate scenarios where the dump lever input triggers automatic lowering while the articulated hauler’s dump body carries a load. This variation assesses the system’s ability to manage the automatic lowering process with the added weight.
– Automatic Lowering with Load Shifting: Introduce scenarios where the load in the dump body shifts dynamically during the automatic lowering process. This variation tests the system’s adaptability to sudden changes in load distribution.

– Sequential Automatic Lowering: Test the system’s performance when executing sequential automatic lowering operations, including raising, automatic lowering, and raising again. This variation evaluates the system’s accuracy and stability during complex sequences.

– Automatic Lowering Interruption and Resumption: Simulate scenarios where the automatic lowering process is interrupted and resumed. This variation assesses the system’s ability to handle interruptions and continue the automatic lowering process.

– Automatic Lowering with Varying Dump Body Angle: Evaluate how the system responds after initiating the automatic lowering process at different dump body angles. This variation ensures the system maintains stability and control during automatic lowering at various positions.

– Automatic Lowering on Different Terrain: Introduce scenarios where the ar-

Figure 5.10: Automatic Lowering
articulated hauler is on different types of terrain while performing automatic lowering. This variation assesses the system’s ability to manage the automatic lowering process on various surfaces.

– Automatic Lowering Speed: Test the system’s performance when executing the automatic lowering process at different speeds. This variation evaluates the system’s responsiveness and accuracy under varying speed conditions.

– Automatic Lowering with Load Compaction: Simulate scenarios where the load compacts during the automatic lowering. This variation tests the system’s ability to handle load compression and adjust the automatic lowering process accordingly.

5.2.2.4 Emergency Stop

• Scenario: Simulate an emergency where the driver issues a hard stop command. (see Figure 5.11)

• Variation:

  – Immediate Emergency Stop: Test the system’s response to an emergency stop command after stopping all actions and movements. This variation assesses the system’s ability to quickly and effectively respond to an emergency.

  – Emergency Stop During Dumping: Simulate scenarios after initiating the emergency stop command while the dump body is in motion. This variation evaluates the system’s ability to safely and promptly stop all actions, including dumping and movement.

  – Emergency Stop with Load: Introduce scenarios where the articulated hauler’s dump body carries a load during an emergency stop. This variation assesses the system’s capability to stop and prevent road hazards.

  – Sequential Emergency Stops: Test the system’s performance when executing sequential emergency stop commands, including resuming normal operations after each stop. This variation evaluates the system’s responsiveness and stability during repeated emergency stop scenarios.

  – Emergency Stop Recovery: Simulate scenarios where the system recovers from an emergency stop and resumes normal operations. This variation assesses the system’s ability to reset and continue functioning after an emergency.
– Emergency Stop at Different Dump Body Angles: Evaluate how the system responds to an emergency stop command at various dump body angles. This variation ensures the system stops safely and stabilizes the load in different positions.

– Emergency Stop in Different Terrain: Introduce scenarios where the articulated hauler is on various terrain types while performing an emergency stop. This variation assesses the system’s ability to stop safely and prevent potential accidents or instability.

– Emergency Stop Speed: Test the system’s performance when initiating an
emergency stop at different speeds. This variation evaluates the system’s quick halt operations capability, even at varying speeds.

- Emergency Stop During Automatic Lowering: Simulate scenarios where the automatic lowering process is ongoing before sending the emergency stop command.

### 5.2.2.5 Unseated Driver

- **Scenario:** Testing the system’s behavior when the seat sensor detects an unseated driver. (see Figure 5.12)

- **Variation:**
  
  - Unseated Driver During Dumping: Simulate scenarios where the driver becomes unseated while the automated articulated hauler dumping system is in the raising process, lowering, or holding the dump body. This variation evaluates how the system responds to sudden changes in the driver’s presence during critical operations.
  
  - Unseated Driver During Automatic Lowering: Test the system’s reaction when the driver becomes unseated while the automatic lowering functionality is in progress. This variation assesses the system’s ability to handle safety concerns and respond appropriately when the driver is not seated.
  
  - Unseated Driver During Load Compaction: Introduce scenarios where the driver becomes unseated while the load in the dump body is compacting. This variation tests the system’s capability to safely manage the load compaction process while the driver is not seated.
  
  - Unseated Driver During Hard Stop: Simulate scenarios where the driver becomes unseated while a hard stop command is in progress. This variation evaluates the system’s response to safety commands when the driver is not seated.
  
  - Unseated Driver during User Interaction: Test the system’s behavior when the driver becomes unseated during user interaction with the Machine Interface, such as inputting commands or monitoring the dumping process. This variation assesses the system’s ability to handle unexpected events during user engagement.
Figure 5.12: Unseated Driver

- Unseated Driver Recovery: Evaluate how the system recovers from an unseated driver scenario. This variation assesses the system’s ability to resume normal operations once the driver seats again without compromising safety or stability.

- Unseated Driver Alarm and Notification: Test the system’s capability to issue alarms or notifications when the driver becomes unseated. This variation alerts relevant parties about the system safety concern.

- Unseated Driver Impact on Dumping Process: Simulate scenarios where the driver becomes unseated at different stages of the dumping process. This variation assesses how the system manages the dumping process’s interruption and continuation when the driver’s presence changes.

- Unseated Driver Response Time: Evaluate the system’s response time to detect and react to an unseated driver situation. This variation assesses the system’s efficiency in identifying safety risks and taking preventive measures.

- Unseated Driver Safety Protocol: Test the system’s adherence to safety protocols when the driver is not seated. This variation follows the system’s appropriate safety measures to prevent accidents or unsafe conditions.
5.2.2.6 Speed Limit Activation

- **Scenario**: Triggering the speed limit activation when the dump body is not in the frame. (see Figure 5.13)

- **Variation**:
  - No Load Speed Limit: Simulate scenarios where the articulated hauler’s dump body is empty, and the speed limit is not activated. This variation establishes a baseline for the articulated hauler’s speed without load-related limitations.
  - Load-Dependent Speed Limit: Test the system’s response to load-dependent speed limits, where the limit is activated only when the dump body carries a load. This variation adjusts the speed limit based on the load’s weight.
  - Load Distribution Impact on Speed Limit: Introduce scenarios where the load distribution affects the activation of the speed limit. For instance, the speed limit might activate if the load distribution is imbalanced. This variation assesses the system’s ability to detect load distribution changes accurately.
  - Dynamic Load Shift and Speed Limit: Simulate scenarios where the load shifts dynamically during the dumping process, leading to the activation of the speed limit. This variation tests the system’s coordination between load changes and speed limit activation.
  - Speed Limit Deactivation: Test the system’s response when the conditions that triggered the speed limit activation change, leading to its deactivation. This variation ensures system can adjust the speed limit based on changing conditions.

![Figure 5.13: Speed Limit Activation](image-url)
– Sequential Speed Limit Activation: Test the system’s performance when executing sequential actions that trigger speed limit activation, such as raising the dump body, automatic lowering, and resuming normal operations. This variation evaluates the system’s accuracy and stability during complex sequences.

– Speed Limit Activation on Different Terrain: Introduce scenarios where the articulated hauler is on various types of terrain while the speed limit is activated. This variation assesses the system’s ability to manage speed limits on different surfaces.

– Speed Limit Adjustment with Load Compaction: Simulate scenarios where load compaction affects the activation or adjustment of speed limits. This variation tests the system’s ability to consider load compression in speed limit calculations.

– Speed Limit Activation Based on Dump Body Angle: Evaluate how the system activates speed limits based on different dump body angles. This variation ensures the system adjusts speed limits according to the dump body’s position.

5.2.2.7 User Interaction

• Scenario: Simulate user interactions with the Machine Interface during dumping. (see Figure 5.14)

• Variation:

  – Valid User Input: Simulate scenarios where users provide valid inputs to the Machine Interface, such as selecting a specific dumping action or viewing the dump body angle. This variation confirms that the system accurately interprets and responds to valid user commands.

  – Invalid User Input: Test the system’s response when users provide invalid or incorrect inputs to the Machine Interface. This variation assesses the system’s ability to handle and communicate errors or incorrect inputs.

  – User Interface Responsiveness: Evaluate the responsiveness of the Machine Interface when users interact with it. This variation confirms that the interface provides timely feedback and updates based on user actions.

  – Simultaneous User Interactions: Simulate scenarios where multiple users interact with the Machine Interface simultaneously, providing different inputs or commands. This variation tests the system’s ability to manage concurrent user interactions without errors or conflicts.

  – User Interface Navigation: Test the ease of navigation within the Machine Interface, such as switching between different functionalities or viewing system
status. This variation assesses the intuitiveness and user-friendliness of the interface design.

- User Feedback and Notifications: Evaluate how the Machine Interface communicates system status, feedback, and notifications to users. This variation confirms that users receive clear and informative messages about the system’s actions and conditions.

- User Input during Function Execution: Simulate scenarios where users provide input to the Machine Interface while a dumping function is in progress. This variation tests the system’s ability to handle user input without interrupting or compromising ongoing operations.

- User Interaction with Dump Body Position: Test how the Machine Interface displays and communicates the dump body position to users. This variation ensures that users can accurately understand and interpret the dump body angle information.

- User Interaction under Varying Conditions: Introduce scenarios where users interact with the Machine Interface under different environmental conditions, such as adverse weather or low lighting. This variation assesses the interface’s readability and usability in challenging situations.
• **Scenario**: Testing the system’s behavior at extreme dump body angles. (see Figure 5.15)

• **Variation**:

  – Maximum Dump Angle: Simulate scenarios where the dump body raises to its maximum angle during the dumping. This variation evaluates the system’s ability to control and stabilize the dump body at an extreme angle.

  – Minimum Dump Angle: Test the system’s response when the dump body lowers to its minimum angle during the dumping. This variation assesses the system’s control and stability at the lowest dump angle.

  – Rapid Angle Changes: Introduce scenarios where the dump body angle changes rapidly and repeatedly during the dumping process. This variation tests the system’s ability to adapt and stabilize the dump body under dynamic angle variations.

  – Angle Hold and Release: Simulate scenarios where the dump body is raised to an extreme angle and then released suddenly. This variation assesses the system’s responsiveness and stability when transitioning from extreme angles.

  – Angle Variation with Load Shift: Test the system’s performance when the dump body angle changes drastically while the load shifts within the dump body. This variation evaluates the system’s ability to maintain control over the load and dump body angle simultaneously.

  – Extreme Angle on Different Terrain: Introduce scenarios where the articulated hauler is on various types of terrain while executing extreme dump angles.
This variation assesses the system’s stability and control over the dump body angle on different surfaces.

- **Extreme Angle with Load Compaction**: Simulate scenarios where the load compacts while the dump body is at an extreme angle. This variation tests the system’s ability to manage load compression and adjust the dump body angle accordingly.

- **Sequential Extreme Angle Changes**: Test the system’s performance when executing sequential extreme angle changes, including raising to a maximum angle, lowering to a minimum angle, and returning to a moderate angle. This variation evaluates the system’s accuracy and stability during complex sequences.

- **Extreme Angle under Load**: Simulate scenarios where the dump body is at an extreme angle while carrying a load. This variation assesses the system’s ability to manage stability and control under extreme angle conditions with added weight.

- **Extreme Angle Speed**: Evaluate the system’s response to extreme dump angles at different speeds. This variation assesses how the system handles extreme angle changes with varying dumping speeds.

### 5.2.2.8 Sensor Malfunction

- **Scenario**: Simulate a scenario where the dump body angle sensor fails. (see Figure 5.16)

- **Variation**:

  - **Dump Lever Sensor Failure**: Simulate a scenario where the dump lever sensor fails to provide accurate input signals. This variation evaluates how the system reacts to incorrect or absent input from the dump lever sensor.

  - **Dump Body Angle Sensor Inaccuracy**: Introduce scenarios where the dump body angle sensor provides inaccurate measurements of the dump body’s position. This variation assesses the system’s ability to compensate for sensor inaccuracies.

  - **Seat Sensor Error**: Test the system’s response when the seat sensor fails to detect the driver’s presence accurately. This variation evaluates the system’s ability to handle incorrect or missing input from the seat sensor.

  - **Load Sensor Anomaly**: Simulate scenarios where the load sensor provides inconsistent or erroneous measurements. This variation assesses the system’s adaptability to unexpected input from the load sensor.
Figure 5.16: Sensor Malfunction

- Communication Disruption: Introduce disruptions in communication between the ECUs, sensors, and the machine interface. This variation evaluates how the system handles temporary communication failures and restores normal communication.

- Sensor Fusion Failure: Test the system’s response when sensor fusion algorithms fail to integrate data accurately from multiple sensors. This variation assesses the system’s ability to handle conflicting or inconsistent sensor data.

- Unresponsive Sensors: Simulate scenarios where sensors become unresponsive or fail to provide input signals. This variation evaluates the system’s ability to detect and handle sensor failures.

- Intermittent Sensor Failures: Introduce intermittent sensor failures, where sensors provide correct input at one time and incorrect input at other times. This variation assesses the system’s resilience in facing unpredictable sensor behavior.

- Sensor Cross-Talk: Test the system’s response to sensors’ cross-talk or interference, leading to distorted input signals. This variation evaluates the system’s ability to filter out noise and differentiate valid signals.

- Sensor Malfunction Recovery: Simulate corrected sensor malfunctions during the operation. This variation assesses the system’s ability to adapt to changing sensor conditions and resume normal functionality.

5.2.3 Boundary and Stress Testing

Boundary testing focuses on testing the system behavior at the boundaries or edges of valid input ranges that help identify vulnerabilities and errors that might arise due to input values at these critical points. Table 5.1 shows BVA test cases for Dump Angle and
Weight control. Table 5.2 shows Stress Testing test cases.

### 5.2.3.1 Boundary Value Analysis

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Description</th>
<th>Input</th>
<th>Expected Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Dump Angle Boundary Test</td>
<td>Set the dump angle to its minimum valid value</td>
<td>0 degrees</td>
<td>System controls dump body angle stably at a minimum</td>
</tr>
<tr>
<td>Maximum Dump Angle Boundary Test</td>
<td>Set the dump angle to its maximum valid value</td>
<td>60 degrees</td>
<td>System controls dump body angle stably at a maximum angle</td>
</tr>
<tr>
<td>Dump Angle Within Valid Range Test</td>
<td>Set the dump angle within the valid range</td>
<td>20 degrees</td>
<td>System maintains dump body angle within specified range</td>
</tr>
<tr>
<td>Dumping with Minimum Load Weight Boundary Test</td>
<td>Initiate dumping load just above minimum valid weight</td>
<td>Minimum + $\varepsilon^4$</td>
<td>System accurately handles dumping operation with stability</td>
</tr>
<tr>
<td>Dumping with Maximum Load Weight Boundary Test</td>
<td>Initiate dumping with load just below maximum valid weight</td>
<td>Maximum - $\varepsilon$</td>
<td>System accurately handles dumping operation stability.</td>
</tr>
</tbody>
</table>

---

3Minimum and Maximum load weight can vary in different vehicles

4Epsilon $\varepsilon$ represents a small quantity, often approaching zero but not quite reaching it. It is commonly used in contexts to express a small change, deviation, or tolerance.
## 5.2.3.2 Stress Testing

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Description</th>
<th>Input</th>
<th>Expected Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous dumping</td>
<td>Perform uninterrupted dumping under maximum load</td>
<td>Continuous dumping operations</td>
<td>System dumps without crashes or slowdowns</td>
</tr>
<tr>
<td>Rapid dumping angle</td>
<td>Rapidly change dump body angle between extremes</td>
<td>Rapid angle changes</td>
<td>System responds quickly and accurately to angle changes</td>
</tr>
<tr>
<td>Load compaction</td>
<td>Simulate load compaction during rapid dumping</td>
<td>Load compaction during dumping</td>
<td>System handles load compaction while maintaining stability</td>
</tr>
<tr>
<td>High-speed dumping</td>
<td>Dump at maximum speeds with different loads</td>
<td>High-speed dumping with load variations</td>
<td>System dumps on high-speed without compromising stability</td>
</tr>
<tr>
<td>User interaction</td>
<td>Initiate dumping while interacting with Interface</td>
<td>Simultaneous dumping and user input</td>
<td>System handles user inputs without compromising dumping</td>
</tr>
<tr>
<td>Sequential dumping</td>
<td>Perform sequences of dumping, lowering, raising</td>
<td>Rapid dumping actions sequence</td>
<td>System executes each sequence action</td>
</tr>
<tr>
<td>Load variation dumping</td>
<td>Dump with varying load weights between operations</td>
<td>Varying load weights</td>
<td>System adapts to load variations while maintaining stability</td>
</tr>
<tr>
<td>Dumping under communication disruption</td>
<td>Initiate dumping with communication disruptions</td>
<td>Dumping with communication interruptions</td>
<td>System recovers from communication disruptions and continues dumping</td>
</tr>
<tr>
<td>Load shifting dumping</td>
<td>Introduce dynamic load shifting during dumping</td>
<td>Dumping with load shifting</td>
<td>System adapts to load shifts and maintains stability</td>
</tr>
</tbody>
</table>
5.2.4 Integration Testing

Here are the integration test cases and techniques for the integrated ECUs mentioned in Section 4.2:

- Testing Technique: Black-box Testing
- Equivalence Partitioning Test Cases

Table 5.3 and 5.4 show a list of test cases for functional and safety requirements mentioned in Sections 4.2.1 and 4.2.2 and expected results corresponding to different inputs.

### 5.2.4.1 Integration Testing - Functional Requirements

<table>
<thead>
<tr>
<th>Equivalence Class</th>
<th>Test Input</th>
<th>Expected Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dump Lever Analog Input - Valid Values</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low Input (1-2)</td>
<td>1, 2</td>
<td>System responds correctly to low input values</td>
</tr>
<tr>
<td>Medium Input (3)</td>
<td>3</td>
<td>System responds correctly to medium input value</td>
</tr>
<tr>
<td>High Input (4-5)</td>
<td>4, 5</td>
<td>System responds correctly to high input values</td>
</tr>
<tr>
<td><strong>Dump Lever Analog Input - Invalid Values</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Negative Input</td>
<td>-1, -2</td>
<td>System handles negative input values appropriately</td>
</tr>
<tr>
<td>Out of Range Input</td>
<td>6, 10</td>
<td>System handles out-of-range input values</td>
</tr>
<tr>
<td>Non-Numeric Input</td>
<td>&quot;abc&quot;, special characters</td>
<td>System handles non-numeric input</td>
</tr>
<tr>
<td><strong>ECU1 Communication - Valid Responses</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Successful Communi-</td>
<td>Valid analog input</td>
<td>System communicates successfully</td>
</tr>
<tr>
<td>cation with Hold Command</td>
<td>Analog input for hold</td>
<td>System communicates hold command</td>
</tr>
</tbody>
</table>
Communication with Raise Command

ECU1 Communication - Invalid Responses

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invalid Data Format</td>
<td>Incorrectly formatted input</td>
<td>Handles invalid data format</td>
</tr>
<tr>
<td>Incomplete Data</td>
<td>Missing components in input</td>
<td>Handles incomplete data</td>
</tr>
<tr>
<td>Unrecognized Command</td>
<td>Invalid input command</td>
<td>Handles unrecognized command</td>
</tr>
</tbody>
</table>

ECU2 and ECU3 Interaction - Normal Operation

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valid Input for Dump Body Movement</td>
<td>Valid analog input</td>
<td>Responds correctly to dump body movement</td>
</tr>
<tr>
<td>Valid Input for Automatic Lowering</td>
<td>Analog input for lowering</td>
<td>Responds correctly to automatic lowering</td>
</tr>
</tbody>
</table>

ECU2 and ECU3 Interaction - Boundary Conditions

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
<th>System Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum Angle Boundary</td>
<td>Analog input for minimum angle</td>
<td>Handles dump body lowering to minimum angle</td>
</tr>
<tr>
<td>Maximum Angle Boundary</td>
<td>Analog input for maximum angle</td>
<td>Handles dump body raising to maximum angle</td>
</tr>
</tbody>
</table>

Table 5.3: Test Cases for Functional Requirements

5.2.4.2 Integration Testing - Safety Requirements

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Equivalence Class</th>
<th>Test Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seat Sensor - Driver Seated</td>
<td>Driver Seated</td>
<td>Test with seat sensor indicating a seated driver</td>
</tr>
<tr>
<td>Seat Sensor - Driver Unseated</td>
<td>Driver Unseated</td>
<td>Test with seat sensor indicating an unseated driver</td>
</tr>
<tr>
<td>Speed Limit Activation - Body Not in Frame</td>
<td>Body Not in Frame</td>
<td>Test with analog input indicating body not in frame</td>
</tr>
<tr>
<td>Speed Limit Activation - Body in Frame</td>
<td>Body in Frame</td>
<td>Test with analog input indicating body in frame</td>
</tr>
</tbody>
</table>

Table 5.4: Test Cases for Safety Requirements
5.2.5 User Interaction Testing

User interaction test cases in this scenario focus on testing system response when the machine interface sends hard stop command. Since the machine interface sends a specific command, we’ll create test cases to verify system’s behavior and responsiveness as shown in Table 5.5.

<table>
<thead>
<tr>
<th>Test Case Description</th>
<th>Command/Input</th>
<th>Expected Outcome</th>
</tr>
</thead>
<tbody>
<tr>
<td>The machine interface sends a hard stop while the dump body is in motion</td>
<td>Command: Hard Stop</td>
<td>System responds by stopping all actions and movements instantly.</td>
</tr>
<tr>
<td>The machine interface sends a hard stop while the dump body is lowering</td>
<td>Command: Hard Stop</td>
<td>System responds by stopping the lowering process and maintaining stability.</td>
</tr>
<tr>
<td>The machine interface sends a hard stop while the dump body is raising</td>
<td>Command: Hard Stop</td>
<td>System responds by stopping the raising process and maintaining stability.</td>
</tr>
<tr>
<td>The machine interface sends a hard stop while the dump body is holding</td>
<td>Command: Hard Stop</td>
<td>System responds by stopping the dump body at its current angle.</td>
</tr>
<tr>
<td>The machine interface sends a hard stop while the dump body is at an extreme angle</td>
<td>Command: Hard Stop</td>
<td>System responds by stopping all actions and movements at the extreme angle.</td>
</tr>
<tr>
<td>The machine interface sends a hard stop while the dump lever provides low analog input</td>
<td>Analog Input: Low</td>
<td>System responds by stopping all actions and movements instantly.</td>
</tr>
<tr>
<td>The machine interface sends a hard stop while the dump lever provides medium analog input</td>
<td>Analog Input: Medium</td>
<td>System responds by stopping all actions and movements instantly.</td>
</tr>
<tr>
<td>The machine interface sends a hard stop while the dump lever provides high analog input</td>
<td>Analog Input: High</td>
<td>System responds by stopping all actions and movements instantly.</td>
</tr>
</tbody>
</table>
The machine interface sends a hard stop when the seat sensor detects a seated driver

Seat Sensor: Driver Seated

System responds by stopping all actions and movements instantly.

The machine interface sends a hard stop when the seat sensor detects an unseated driver

Seat Sensor: Driver’s Unseated

System responds by stopping all actions and movements instantly.

Table 5.5: Test Cases for UI

These user interaction test cases focus on the machine interface sending a hard stop command in various scenarios. They ensure the system responds correctly and promptly to the hard stop command issued by the machine interface, providing a comprehensive assessment of the system’s behavior in these situations.

5.3 Data Collection Overview

This section presents the comprehensive strategy to gather relevant and insightful data for our research project. This strategy involves a structured approach to data collection that aligns with the research objectives and establishes a strong foundation for addressing the research questions.

5.3.1 Data Collection

Three primary sources for data collection were employed: Volvo Software Engineers (supervisors), Bitbucket, and Confluence pages. Each source can help based on its direct relevance and significant contribution to the research context.

Insights and guidance from supervisors during meetings offered valuable expert perspectives by providing necessary data in a shared folder to start the project, which finally improved the qualitative nature of the collected data.

Volvo Bitbucket played a vital role in solving the complexities in the dumping scenario’s implementation and testing. Bitbucket provides a valuable resource for tracing the integration points between ECUs and their corresponding CAN signal names, which cross multiple ECUs. Including code and comprehensive CAN signal documentation streamlined the process of establishing interconnections across various chassis components, facilitating the clear sketch of integrated functional requirements. As a result, this data source significantly enriched the research with precise and in-depth technical insights, impeccably aligned with the research’s technical orientation.
Furthermore, Confluence pages served as a collaborative platform for accessing project-documented system architecture and high-level design through Unified Modeling Language (UML) and System Modeling Language (SysML). This platform can be a valuable resource for extracting comprehensive insights into the overall system structure and design, enhancing the research’s ability to analyze integration aspects across various components.

The data collection process paid careful attention to ethical considerations. When utilizing data from supervisors, the information shared during meetings and discussions remained confidential. Any sensitive or proprietary information was controlled and agreed upon before recording and data usage.

This research maintains its integrity and upholds the ethical standards for responsible data utilization.

5.3.2 Data Analysis

The collected data from the mentioned sources include a valuable resource for gaining in-depth insights into the dumping scenarios integration within the context of embedded systems. The analysis process involves a content analysis of the data. Content analysis is a fundamental research technique in the social sciences, recognizing that societal understanding hinges on interpreting language and communication in various forms [29]. In the computer science field, it encompasses the systematic examination of textual, visual, or communicative data to extract underlying patterns, semantic meanings, and underlying intentions within their digital contexts. Separating from widely used methodologies in natural sciences, which may concentrate on quantitative aspects, content analysis in computer science delves into the nuanced ways language and digital communication influence technology-driven phenomena. Reviewing the links from sources to system architecture and tool installation identifies stakeholders’ functional and safety requirements. This qualitative analysis sheds light on integration points that need testing.

Moving to the technical data derived from the Volvo Bitbucket and Confluence pages, a more technical analysis was performed. Code implementations and CAN signal documentation helped integrate qualitative insights from meetings with technical details from data repositories, laying the groundwork for creating integration test cases. This analysis aimed to identify bottlenecks, compatibility issues, and data flow pathways that impact successful integration testing. By examining the codebase and its associated documentation, this study provides a comprehensive understanding of technical complexities, enabling the identification of potential integration gaps and testing requirements.

Combining qualitative insights from meetings and technical details from data repositories offers a comprehensive understanding of the integration landscape, thus forming the foundation for creating efficient integration test cases.
6 Results

6.1 RQ1

"What are the suitable methods for conducting integration testing at the ECU level?"

Integration testing at the ECU level is a critical development phase that ensures the proper functioning and reliability of the integrated system. Several testing methods are particularly suitable for ECU integration testing to achieve these goals:

1. Equivalence Partitioning: This method categorizes input values into equivalence classes and tests each class representative values. It aids in validating inputs from various sensors and actuators that contribute to the ECU’s operation. Testing input values from different ranges ensures that the ECU accurately process input signals and responds appropriately.

2. Boundary Value Analysis: Testing values at the valid ranges boundaries uncovers potential issues that may arise due to boundary conditions. It ensures that the ECU functions correctly and transitions smoothly between different states. For example, testing inputs below and above the valid ranges helps identify unexpected behavior or transitions.

3. Stress Testing: Subjecting the ECU to extreme conditions, such as rapid angle changes, continuous operations, and varying loads, helps identify potential weaknesses and assess the system’s resilience under challenging scenarios. Stress testing provides insights into the ECU’s behavior under demanding conditions, ensuring robustness and stability.

6.2 RQ2

"Can a single testing method provide sufficient support for fault detection and coverage at the ECU integration level?"

While individual testing methods contribute valuable insights, relying on a single testing method may not provide sufficient support for comprehensive fault detection and coverage at the ECU integration level. Every testing method has strengths and is tailored to the specific system behavior aspect. Utilizing a combination of methods is essential for ECU integration’s robustness examination.

For instance, equivalence partitioning and boundary value analysis identify input-related errors, ensuring that the ECU correctly processes various inputs. Stress testing and testing under extreme angles expose stability and control weaknesses under challenging conditions. A holistic approach includes integrating multiple methodologies to help uncover a wide range of faults, vulnerabilities, and unexpected interactions between components.
6.3 RQ3

How effective is ECU integration level testing in reducing feedback-loop?

Testing at the ECU integration level significantly reduces delays in the feedback loop by identifying and addressing issues early in the development process. By detecting faults at the ECU level before they propagate to higher levels of integration or deployment, this testing phase reduces the likelihood of encountering critical defects during later stages. Fixing issues earlier enhances development efficiency and reduces the time spent in feedback loops between the development, testing, and validation phases.

Effective ECU integration testing enhances the reliability and stability of the overall system, reducing the likelihood of costly rework or redesign. Identifying fewer critical issues in later stages makes the development process smoother, project timelines more predictable, and enhances the overall quality of the integrated system.

In conclusion, a comprehensive approach that combines multiple testing methods at the ECU integration level ensures robustness, safety, and stability. By identifying a broader range of issues, this approach contributes to more efficient development cycles, ultimately reducing delays in the feedback loop and enhancing the overall quality of the integrated system.
7 Analysis

In this chapter, the analysis focuses on evaluating how our implementation addresses stakeholder ASRs. Evaluating our implementation against these criteria, verify its compatibility, effectiveness, and reliability in meeting the Volvo CE system’s critical architectural needs and performance expectations despite not being directly integrated. Rigorous testing and assessment are essential for validating our approach.

1. Real-time Responsiveness: The testing approach addressed this requirement by including stress tests simulating rapid angle changes and high-speed dumping scenarios. These stress tests assessed the system’s ability to respond promptly to user inputs and sensor data changes.

2. Reliability and Safety: The stress tests in the testing approach contribute to the system’s reliability and safety validation.

3. Interoperability with Existing Systems: The integration tests in the approach focused on communication between integrated ECUs.

4. User-Friendly Interface: The testing approach addressed system behavior during user interactions to evaluate the intuitiveness and effectiveness of the machine’s graphical user interface (GUI).

5. Data Logging and Analytics: The approach covered stress tests that involve data logging during various dumping scenarios.

6. Maintainability and Upgradability: The testing approach aligned with this ASR by focusing on integration tests that ensure component compatibility and functionality.

7. Integration with External Sensors and Actuators: The current testing approach primarily addressed integration among provided components.

8. Modularity and Scalability: The testing approach aligned with this ASR by conducting integration tests that verify the interaction between different ECUs and components.

8 Discussion

The testing approach presented in this study describes a comprehensive effort to validate and verify the new implementation of the articulated hauler dumping system aligned with the identified ASRs. Various testing techniques and scenarios ensure that software system architecture and behavior met specified requirements and industry standards.

Comparing our testing approach with related work (Section 1.2), we find alignment in the emphasis on stress testing and scenario-based evaluation. Similar studies have highlighted the importance of subjecting systems to extreme conditions to uncover vulnerabilities and assess their resilience. Our stress tests, simulating rapid angle changes, high-speed dumping, and communication disruptions, confirm these findings by showcasing the responsiveness and robustness of the system under challenging scenarios. However, enhancements are required to address ASRs. Although stress tests provide valuable insights, a broader range of fault injection and failure mode tests are necessary to validate the system reliability and fault tolerance mechanisms. In addition, the absence of explicit security and authentication testing poses a potential gap in addressing ASRs related to security. Incorporating security testing methods such as penetration testing and vulnerability assessment is crucial for identifying and mitigating potential security risks. The adaptability of the approach to changes and upgrades is a strongly point to emphasize integration and maintainability testing. However, to further enhance this aspect, introducing tests that simulate version updates and patches would provide a more realistic assessment of system upgradability without disrupting ongoing operations.

To comprehensively evaluate the system’s capabilities and align them with ASRs, it is advisable to augment testing by including security, usability, power efficiency, hardware failure, energy consumption, and regulatory compliance tests. Ensuring conformity with industry regulations and standards is vital to legal compliance and adherence to requirements. Real-time responsiveness can be enhanced by incorporating tests that measure system latency at various stages of the dumping process. Additionally, bolstering the approach with fault injection and failure mode simulation tests; assessing energy consumption, hardware reliability, and interoperability with external systems; and conducting tests to validate data accuracy, completeness, and retrieval for diagnostics and performance evaluation would fortify the assessment.

In this thesis, our focus is on articulated hauler-dumping scenarios to demonstrate the effectiveness of our testing approach. However, the versatility of our methodology extends its potential application to a wide array of domains beyond hauling. The testing methodologies and tools employed can be generalized and adapted for diverse sectors such as aerospace, railway, and healthcare. For instance, our approach can be used in aerospace to test automated landing procedures, navigation systems, and communication protocols. Similarly, within the railway industry, this approach can be applied to safety testing for automated braking systems, monitoring train movements, and assessing track
condition monitoring systems. Furthermore, the healthcare sector offers prospects for
testing and validating automated drug dispensing systems, patient-monitoring devices,
and scheduling and managing medical resources efficiently. The adaptability and cus-
tomization capabilities of our approach to meet the unique requirements of each domain
are testaments to its broad applicability. Interdisciplinary collaboration could further en-
hance its potential by combining expertise from different fields, leading to innovative
testing approaches applicable across various industries.

Incorporating OpenModelica into our articulated hauler testing presents immense po-
tential, but it is essential to recognize its limitations. OpenModelica’s intricate nature can
pose a steep learning curve, demanding substantial time and effort for proficiency. Partic-
ularly for our sizable articulated hauler system, simulation times may become a hindrance,
impacting the efficiency of our testing and validation processes. Striking the right balance
between accuracy and simulation speed is crucial, often involving trade-offs. Moreover,
while OpenModelica offers visualization capabilities, they might be limited to highly de-
tailed or specialized representations, necessitating additional visualization tools. Large
complex models as hauler needs efficient model handling, and computational load during
simulations become a priority. In addition, real-time simulation optimization and seam-
less integration with external systems could be areas of improvement. Tailored pre-built
libraries for our specific domain, heavy machinery, might be lacking, potentially requiring
substantial custom modeling efforts. Understanding and mitigating these limitations will
be pivotal in leveraging OpenModelica effectively for broader use cases beyond articu-
lated hauler scenarios, including aerospace, railway, and health applications.
9 Conclusions

In conclusion, the implementation and testing of the articulated hauler dumping system have provided valuable insights into its performance and alignment with ASRs. The comprehensive testing approach, encompassing boundary, stress, integration, and user interaction tests, sheds light on the system’s strengths and potential areas for improvement. Notably, the testing approach effectively addressed various ASRs, showcasing the system’s ability to handle stress conditions, interact with integrated ECUs, and respond to user commands. However, there is room for enhancement by refining the approach to encompass security, usability, power efficiency, and regulatory compliance aspects representing directions for future work.

Several avenues for future work emerge from this study. Enhancing security testing through penetration testing and vulnerability assessment is crucial for fortifying the system against potential threats. The usability testing integrated with user feedback collection could optimize the user interface and overall user experience. Exploring power-efficient components, conducting power consumption tests, and integrating regulatory compliance tests aligned with industry-specific regulations would contribute to fulfilling significant ASRs. Furthermore, extending the testing scope to cover different failure modes and exploring advanced simulation techniques could provide a more comprehensive evaluation of the system’s reliability and fault tolerance mechanisms.

Moreover, refining the documentation and learning resources for OpenModelica is essential to reduce the learning curve associated with its complexity. Implementing the tested approach on an articulated hauler system and evaluating its impact on cost, feedback loop reduction, and time complexity represent notable future directions. This practical deployment could yield valuable insights into the framework’s real-world performance and feasibility, potentially leading to further refinements and optimizations for industrial use.

This bachelor thesis project focused on the integration testing of an articulated hauler dumping system for Volvo CE, tackling challenges related to ensuring seamless system operation and integration with Volvo CE’s existing infrastructure. The high-level idea of the proposed work involved validating the system’s functionalities and performance under various stress conditions. Furthermore, the potential generalization of this work to benefit other domains, such as automotive and aerospace, underscores the versatility and applicability of the testing methodologies developed. Future work should prioritize enhancing security measures to fortify the system and exploring broader applications in different industry domains.
References


[27] [Online]. Available: https://modelica.org/libraries.html
