Bridging the Gap: Socio-Technical Perspectives on Software Testing and Automation in Manufacturing

Jasmin Jakupovic[1,*], Joel Johansson[1,2,†], Rudy Matela[1,†] and Johan Lidén Eddeland[3,†]

[1]School of Engineering, Jönköping University, 553 18 Jönköping Sweden
[2]Thule Sweden AB, 335 73 Hillerstorp Sweden
[3]Combitech AB, 583 30 Linköping Sweden

Abstract

The integration of software development with traditional product development in manufacturing industries is increasingly critical. This paper examines software testing and test automation in Swedish manufacturing companies through a *socio-technical* perspective, identifying key trends, challenges, and best practices. A study of seven companies reveals that in-house development fosters higher test automation levels, improving product quality and reducing lead times, whereas outsourced development struggles with synchronisation and automation.

The study underscores the need for hybrid models blending agile methodologies with traditional stage-gate processes to address *technical* challenges—such as tool integration and automation scalability—and *social* challenges, including organisational adaptation and cross-functional collaboration. Software testing and automation are not solely technical processes but are deeply influenced by organisational structures and decision-making.

Findings highlight the potential of automated testing to enhance efficiency, enable early defect detection, and bridge software-hardware collaboration gaps. This paper provides insights for practitioners and researchers to optimise software testing strategies in industrial settings and proposes future research directions to refine socio-technical approaches to test automation and integration.

Keywords

Software Testing, Automated Testing, Software Development, Product Development, Connected Product Development, Cyber-Physical Systems, Agile Stage-gate, Socio-technical Perspectives

1. Introduction

Software engineering drives modern technological progress, significantly transforming traditional product development into connected systems [1]. Devices such as robot vacuum cleaners, lawnmowers, and fitness trackers illustrate this shift (Figure 1). This evolution demands seamless integration between software and hardware, yet manufacturing companies struggle to balance structured, milestone-driven processes [2] with the rapid iteration of agile software development [3].

Software testing and automation ensure reliability and efficiency in connected products. Testing reduces costs and prevents defects [4, 5], while automation improves repeatability and reusability [6]. However, the timing and extent of automation in manufacturing workflows remain unclear, particularly when balancing software, firmware, and hardware development.

Manufacturers traditionally follow rigid development cycles with significant late-stage investments [7]. However, connected products require continuous updates, necessitating adaptable testing strategies. The *socio-technical systems* (STS) perspective [8, 9] highlights the interaction between technology and organisational structures, affecting test automation adoption and efficiency [10]. Companies with in-house development often achieve higher automation maturity, whereas outsourced models face integration challenges.

The 11th International Workshop on Socio-Technical Perspectives in IS (STPIS'25) September 17-18 2025 Skopje, North Macedonia.
*Corresponding author.
†These authors contributed equally.
jasmin.jakupovic@ju.se (J. Jakupovic); joel.johansson@thule.com (J. Johansson); rudy.matela@ju.se (R. Matela); johan.lideneddeland@combitech.com (J. L. Eddeland)
0009-0002-1728-9544 (J. Jakupovic); 0000-0003-1162-724x (J. Johansson); 0000-0001-5302-7096 (R. Matela); 0000-0002-1253-6705 (J. L. Eddeland)

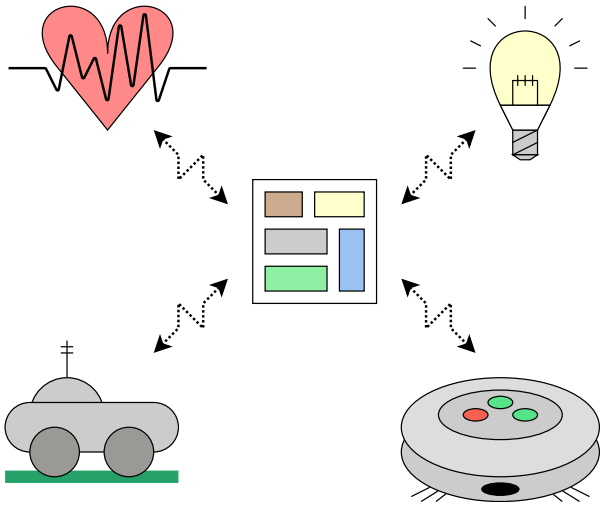

Figure 1: Examples of connected products managed by an app: a smart lamp, a robot vacuum, a robot lawn mower, and a fitness heart rate monitor.

This study examines software testing and automation practices in manufacturing, identifying industry trends and socio-technical challenges. It explores how companies integrate software testing into product development, the stages at which testing occurs, and the impact of automation on product quality and lead times. By addressing these issues, this research provides insights to optimize software testing strategies in industrial settings.

2. Related Work

Software testing in manufacturing industries spans multiple disciplines. This section reviews agile development in manufacturing, software testing practices, and automation trends to understand how these elements integrate into product development.

2.1. Agile in Manufacturing Industries

Manufacturing companies increasingly incorporate digital solutions into their physical products [11, 12]. Agile methodologies have been explored as a means to enhance responsiveness and product quality [13]. While effective at the team level, they do not always reduce lead times due to slower hardware development cycles [14]. Hybrid models combining agile flexibility with structured stage-gate processes have been proposed to balance iterative software development with controlled product development [15, 16, 17].

2.2. Software Test in Industrial Settings

Studies on software testing in industry highlight challenges in methodology adoption, tool integration, and tester expertise [18]. Exploratory Testing (ET) and Interactive Search-Based Software Testing (ISBST) allow domain experts to contribute effectively, even without formal testing knowledge [19, 20]. Software Test Process Improvement (STPI) frameworks aim to enhance test strategies, balancing structured and flexible approaches [21].

Comprehensive literature reviews identify gaps in test management, risk-based testing, and automation scalability [22, 23]. Industry-academic collaborations highlight the practical application of software testing in manufacturing [24, 25]. Regression testing remains a key area of study, focusing on industrial applicability and flaky test mitigation [26, 27]. Despite advancements, significant challenges persist in test automation and legacy system testing [28, 29].

2.3. Levels of Software Test Automation

Automated test execution and generation improve efficiency [30]. Studies demonstrate effective automated unit test generation [31] and test case generation for RESTful APIs [32]. Industrial applications of automation tools face challenges in scalability and reporting [33, 34]. Empirical studies validate the benefits of test automation maturity on product quality [35].

AI-driven test automation enhances efficiency by automating non-functional requirements and prioritizing test cases [36, 37, 38]. Research underscores the importance of adapting AI techniques to industrial needs [39]. Cyber-physical systems introduce security concerns, necessitating semi-automated security analysis [40].

2.4. Socio-Technical Perspectives on Software Testing and Test Automation

Software testing and automation are influenced by organisational structures, team dynamics, and decision-making processes [8, 9]. Socio-technical systems (STS) theory suggests successful technology adoption requires alignment between technical and social components [10]. Companies with strong knowledge-sharing cultures achieve higher automation maturity [41].

Cross-functional collaboration is crucial for integrating test automation into product development [42]. Hybrid agile-stage-gate models are needed to balance flexibility and structured decision-making [14]. Future work should explore how STS principles can optimise test automation strategies in manufacturing environments.

Recent literature on digital infrastructure, organisational ambidexterity, and resistance to digital innovation [43, 44, 45, 46], provides important extensions to classical STS thinking. While these themes were not the primary focus of this study, they represent promising directions for interpreting the integration challenges faced by manufacturing firms transitioning toward software-intensive product development.

3. Knowledge Gap and Research Questions

3.1. Knowledge Gap

As mentioned in the previous section, balancing agility with the rigorous demands of stage-gate methodologies takes time and requires effort. It is obvious that product development (*stage-gate*) tends to follow a strict process, while software development (*agile-scrum*) tends to follow a more flexible process. Despite the substantial research on software testing methodologies and their applications within industrial settings, several critical gaps remain. These gaps primarily pertain to integrating software development processes with traditional product development, particularly in manufacturing companies incorporating firmware and software into their physical products.
Key areas where knowledge is lacking include:

1. **Integration of Agile and Stage-Gate Processes.**
   While there has been considerable research on agile practices in manufacturing industries, the seamless integration of agile methodologies with the traditional stage-gate process remains underexplored. Although prior research has recognised the disconnect between agile software practices and stage-gate-driven product development, there remains insufficient understanding of why these misalignments persist in manufacturing contexts.
   The hybrid models proposed by Cooper and others suggest potential benefits, but empirical validation and industry-specific adaptations are needed. More specifically, for this study, where does software testing fit in the process? We argue that this persistence reflects not merely technical or procedural shortcomings, but deeper socio-technical tensions, such as organisational inertia, legacy culture, and the fragmentation of knowledge across teams. Addressing these issues demands a socio-technical perspective that goes beyond surface-level process integration.

2. **Timing and Extent of Software Testing in Connected Product Development.**
   The literature indicates that testing is critical to software and product development. However, there is a lack of detailed understanding regarding when and how software testing, especially automated testing, is conducted in the context of connected product development. This includes identifying specific stages within the stage-gate process where testing occurs and the extent of automation used.
3. **Challenges and Impact of Automated Testing in Connected Product Development.**
   While the benefits of test automation are well-documented, practical challenges such as tool integration, scalability, and the adaptation of academic tools for industrial use are not sufficiently addressed. This gap is particularly pronounced in manufacturing companies where the complexity of connected products necessitates sophisticated testing approaches. Also, the influence of automated testing on the overall product development cycle, particularly in reducing lead times and enhancing product quality, requires further investigation. Existing studies highlight the potential of automation to improve efficiency, but detailed empirical data on its impact in industrial settings, especially in manufacturing, is sparse.

3.2. Research Questions

To address these gaps, the following research questions are proposed:

> ***RQ 1**: How does software development fit in the product development process in manufacturing companies?*

> ***RQ 2**: At what stages of the connected product development process is software testing, particularly automated testing, conducted in manufacturing companies?*

> ***RQ 3**: What is the impact of automated testing on the product development cycle in manufacturing companies?*

4. Methodology

Unlike many software engineering problems, which require a quantitative approach [47], this research focuses more on the qualitative aspects. Inspired by the work of [48], we proceed with qualitative methods as they've proven more effective in answering broader research questions [43]. The knowledge obtained through the qualitative study conducted in this research will serve as a foundation towards more quantitative studies on the field of software test automation. Our research process, as presented in Fig. 2, explains the approach when conducting our research, where we start by setting up the study, identifying and pinpointing the needs, conducting the interviews, transcribing and coding data, iterating on analysing findings and discussing, and finally concluding.

This research results from a collaboration between the School of Engineering at Jönköping University and one of the case companies in this research. Tacit knowledge about the company and its products and processes is therefore present. This knowledge is based on the long-term, close working relationship between the researcher and the company.

4.1. Participant Selection

Participants for this study were selected from seven manufacturing companies that engage in software development, internally or by outsourcing to dedicated software development teams. Notably, two of the companies in question had external assistance when developing software for their connected

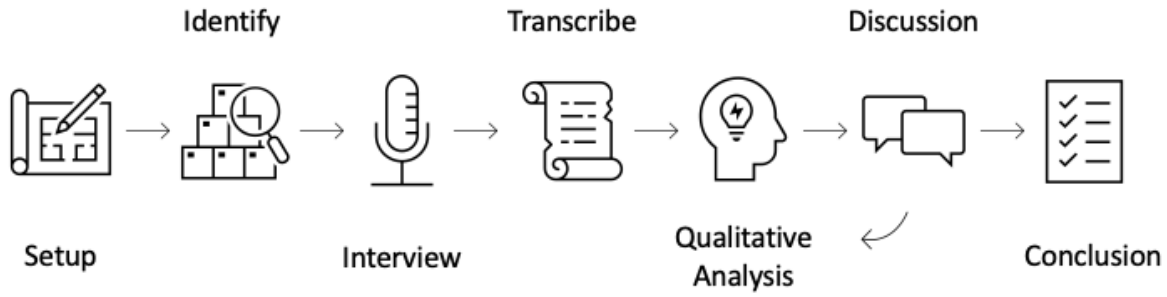

Figure 2: Research Process - From setting up the study to final discussions and conclusions

products. Some participants represent an outsourcing company engaged in the project, providing an external perspective on the software test automation strategies used. To capture an understanding of software test automation strategies within these companies, each company contributed with one to three interviewees for their respective development departments. Top management was initially approached to approve the research focus and help us identify suitable interview candidates. The selection criteria focused on identifying individuals who held management roles with personnel responsibilities, leading teams ranging from tens to hundreds of developers. This approach ensured the participants had a strategic overview and a profound understanding of their organisations' software test automation strategies.

This careful selection of participants from various hierarchical and organisational contexts within the software development lifecycle was designed to elicit insights from those directly involved with or responsible for implementing and overseeing test automation processes. By targeting management-level interviewees, the study aimed to gather data from those most likely to have a holistic view of their company's approach to software test automation, including both strategic and operational dimensions.

4.2. Interview Design

The interview script was developed in collaboration with the collaborating company, based on their experience and challenges, and the research team, which consists of senior academic researchers and industry experts specialising in software development. The group initially identified the theme of the study, which revolves around software testing and test automation in manufacturing companies developing connected products. Guided by this theme, questions were formulated to capture a comprehensive understanding of the software development lifecycle within these companies. To obtain the necessary information, semi-structured interviews were designed following recommendations provided by [49]. The questions were structured to explore key aspects, including when testing is conducted, what elements are tested, and the extent to which testing is automated. This approach ensured that the script effectively addressed the research objectives by eliciting detailed insights into the practices and challenges of software test automation in the targeted industry context.

A set of main questions was identified to capture the overall process, complemented by follow-up questions and probes to dig deeper into the necessary topics. Probes were used at stages where further clarification was needed also to keep the interviewees on track. The interview's main questions and follow-up questions, some of which were translated and asked in Swedish, were:

1. How does the Software Development Processes look at your company?
2. When does Software Development occur in regard to Product Development
   (a) Development of Software
   (b) Development of Firmware
   (c) Common problems between the two
   (d) Reflections

3. Is Software Development done in-house, or is it outsourced?
   (a) If outsourced, what is the outsourcing strategy?
4. When do you conduct Software Testing?
   (a) What are your test strategies?
   (b) Do you utilise automated testing?
   (c) What are your views and approaches regarding test automation?
   (d) What types of tests do you conduct?
5. **Conclusion**: What is the approach to software testing at *your company*, to what extent have you utilised automated testing, and what and when do *you* test?

4.3. Interviews

Semi-structured interviews were used to collect qualitative data that cannot be captured through quantitative measures [50]. The interviews provide insights into participants' experiences, opinions, and feelings about software development practices. Interviews were conducted with one to three suitable candidates. Where agreed upon, interviews were recorded, otherwise, notes were taken in the interview guide during the interviews. The transcripts and interview guide notes were later *coded* to extract values. The coding process extracts values for quantitative variables from qualitative data, which allows us to perform certain quantitative analyses based on the qualitative findings [51]. To ensure the validity and reliability of the study, all conclusions are traceable to specific interviews, which, to ensure integrity, have been pseudonymized.

4.4. Pseudonymization Process

After data collection, each company was assigned a unique pseudonym (e.g., Alpha, Beta, Gamma, etc.). These pseudonyms were used consistently throughout the study to refer to the companies, ensuring no real names or identifying information were included in any documentation or analysis accessible to anyone outside the research team. Any explicit references to specific products, services, or internal projects that could potentially identify the company were redacted from the data, and information such as product names, names of clients, names of partnerships, and names of proprietary technologies or software. Demographic data were aggregated to prevent identification based on workforce characteristics. For example, ranges were used instead of stating the exact number of employees (e.g., "between 100 and 300 employees"). This approach was also applied to departmental sizes and roles within the companies. Indirect identifiers that could lead to the identification of a company through data triangulation were reviewed and altered, including changing or removing any specific examples or case studies provided by the companies during the interviews. All documents with non-pseudonymized data were securely stored with access to the research team only. Multiple research team members cross-checked the pseudonymized data to verify that no inadvertent identifiers remained.

With the implementation of these measures, we ensure that the data used in this study maintains the confidentiality and privacy of the participating companies. This allowed for a thorough and unbiased analysis while upholding ethical standards in research. Using pseudonymization allowed the research team to retain the ability to re-identify the participants if necessary, ensuring the robustness and flexibility of the study.

4.5. Data Analysis

We emphasise that analysing qualitative data in empirical software engineering studies requires rigorous analysis over merely recording impressions and subjective interpretations [51]. A good contextual understanding of each company prior to interviewing is recommended to improve the rigour of qualitative research [52]. In our study, we aim to capture developer experiences, tool and process evaluation, and adaptation of practices. This will be done by using a thematic analysis approach applied to empirical

software engineering [53, 54]. As a first step, we read through the transcripts, familiarising ourselves with the data's depth and nuances. A *grounded theory* method [55] is applied, allowing us to extract statements supported by the data in multiple ways. Grounded theory is used because of its adaptability in these types of studies [56]. An initial set of codes is identified based on the theme of the study and its goals. This set of codes is used throughout the coding process but is not final and can expand depending on the findings. Coded findings are then evaluated to find themes, trends and patterns [51]. These are afterwards reviewed and refined to ensure accuracy in data presentation. Themes are then defined and named to help present coherent findings in a structured manner.

To increase and ensure the study's validity, all findings were discussed within the group of researchers before conclusions were made, following the process presented in Fig. 2.

5. Results & Discussion

In this section, we present our findings based on the interviews conducted with seven companies, each developing one or more connected products. For the reader to better understand the companies, we start by presenting the size and type of company. Afterwards, we identify and present the processes for each company. Lastly, we give an overview of software tests and software test automation in respective companies. After we present the companies and relevant processes, we present trends and patterns in the data.

5.1. Grouping of Companies

The companies were grouped based on their software development approaches, as well as additional characteristics such as the number of employees, types of products, and company type. This is shown on Table 1. Following the pseudonymity guidelines, we used broader employee number ranges to minimise the possibility of company identification. Companies are classified as Original Equipment Manufacturer (OEM) and Contract Manufacturer (CM) companies, with some being both OEM & CM at the same time. Furthermore, classification based on the products being developed across the entire company has been made:

- **Single product** or **Multiple product**: Does the company develop one or more digital products?
- **Single app** or **Multiple app**: Is there one single end-point for the digital product(s); or are there multiple end-points for the digital product(s) to connect to?

In our classification, we combine the type of product(s) being developed with the type of application/system used to connect to the products, providing an interface to the end user as presented in Fig. 3.

Noteworthy is that, in addition to the three cases there is a fourth case where a product may be connected to by multiple applications or be compatible with third-party applications and external integrations. However, our study specifically focuses on the internal development and integration efforts carried out within the companies themselves. Therefore, we limit our analysis to the software applications and system architectures that are directly developed, owned, or maintained by the companies. This allows us to investigate the socio-technical alignment between internal product development and testing practices, without conflating it with external ecosystem complexity, which lies beyond the scope of this study.

It is important to note that companies such as Epsilon, with a large variety of products and employees and branches spread worldwide, can have different approaches to software development depending on the product in question. Companies such as Zeta, which was quite immature with connected products and is just finding its way in the digital product market, ought to mature and further evolve their software development practices. Companies such as Gamma and Theta, with a multiple product and multiple app structure, might also have different practices at different departments developing for different products. These limitations are discussed further in Section 5.6. Therefore, it is hard to

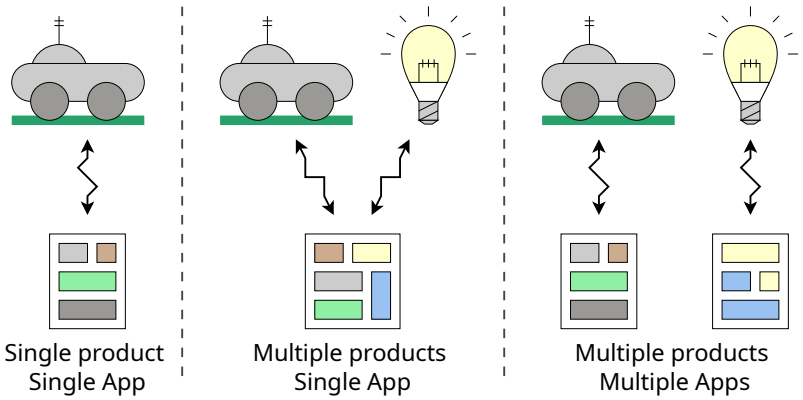

Figure 3: Classification according to single or multiple connected product, and single or multiple app in manufacturing companies

Table 1
Company characteristics, Development approaches, and Extent of test automation

Company	Size (empl.)	Product	Platform	Development	Test automation extent
Alpha	1000-5000	OEM	Mulitple product, single app	In-house	Medium to High
Beta	500-1000	OEM	Mulitple product, single app	In-house	Medium to High
Gamma	10000-50000	OEM	Mulitple product, multiple app	In-house	Medium
Epsilon	100000+	OEM	Mulitple product, multiple app	Hybrid	Low
Zeta	1000-5000	OEM	Single product, single app	Outsourced	Low
Eta	1000-5000	CM	Mulitple product, single app	In-house	Medium
Theta	10000-50000	OEM & CM	Mulitple product, multiple app	In-house	Medium to High

generalise that mid-large and very-large companies function similarly and use the same processes across their entire enterprise. However, from what we concluded throughout our research, this might be the as-is scenario for most development conducted in companies such as Gamma and Theta (mid-large). At the same time, we can not claim the same for Gamma (very-large) due to the complexity of the organisation.

5.2. Software/Firmware Development in Connection to Product Development

For us to be able to find trends and patterns and ultimately draw conclusions about them, it is essential to identify when these companies conduct software/firmware development concerning product development. During our interviews, we tried to pinpoint when software and firmware were developed in connection to the physical product. Furthermore, we wanted to understand the connection between these two processes. In Table 2, we briefly summarise what the connection between the three processes looks like (software, firmware, physical product). With this, we also answer our ***RQ 1****: "How does software development fit in the product development process in manufacturing companies?"*

The results highlight the diversity in development processes:

- Companies following the stage-gate model (e.g., Alpha) align software development closely with product development milestones.
- Hardware-driven companies (e.g., Beta, Gamma) prioritise firmware development early, whereas software development follows later stages.
- Companies with simultaneous software and firmware development (e.g., Theta) treat software as an integral part of the product rather than an add-on.
- Fully outsourced companies (e.g., Zeta) have minimal influence over software development, limiting opportunities for integration and optimisation.

Table 2
Software development process in relevance to the product development process

Company	Development Process
Alpha	In line with stage-gate, following gates closely.
Beta	Hardware-driven development of products, firmware developed earlier; software a bit later.
Gamma	Manufacturing-driven, firmware developed earlier; software developed late. Transitioning to integrated processes.
Epsilon	Firmware developed in-house during product development, software developed for the finalised product.
Zeta	Fully outsourced firmware and software development, working with an existing product.
Eta	Software development integrated late, working with an existing product.
Theta	Simultaneous software and firmware development; product developed as a result of software/firmware development.

5.3. Software Test and Test Automation

This subsection identifies the levels of software testing and software test automation in the seven companies interviewed in this study. Software testing approaches varied significantly among the interviewed companies. Table 1 presents the extent of test automation across companies. The key observations include:

- Companies with mature in-house software development practices (Alpha, Beta, Theta) exhibit greater investment in automated testing.
- Hybrid models (Epsilon) show inconsistencies in automation due to organisational complexity.
- Fully outsourced companies (Zeta) have limited automation due to dependence on third-party vendors.

It is important to note that no formal definition of software test automation was established before the interviews were conducted. Instead, the companies judged the extent of their test automation based on their own notions of testing and test automation. We believe that the definition of software test automation goes in line with the maturity of the companies in regard to software development which allows us to classify the companies from their standpoint and current maturity level. From our data, it was very clear that the majority of our respondents did not really refer to the gates in the product development process. Instead, they have used terminology such as early, middle, and late to emphasise when their contribution begins in the process. It is also crucial to emphasise that different departments come into the process at different stages. For example, hardware-dependent development (firmware development) was introduced sooner than application/system development (software development). Although these findings do not directly map to the second research question, ***RQ 2****: "At what stages of the connected product development process is software testing, particularly automated testing, conducted in manufacturing companies?"*, it does give us a clear picture of when in the stage-gate process start developing software and at what stage of the software development process they conduct testing. Early testing, or, as some noted, quality assurance, is critical to the early detection and prevention of defects in connected products. This knowledge could drastically impact production costs and lead times. Several examples of such issues were presented throughout the interviews, where early testing saved the company from shipping a faulty product to production. These findings also tap into ***RQ 3****: "What is the impact of automated testing on the product development cycle in manufacturing companies?"*

5.4. Socio-Technical Implications of Software Testing in Manufacturing

Software testing and test automation in manufacturing companies are not purely technical activities but are deeply influenced by socio-technical factors. The effectiveness of test automation depends not only on tool selection and integration but also on organisational culture, knowledge-sharing, and cross-functional collaboration [10].

Our findings indicate that:

- Companies with in-house software development align testing with product development cycles more effectively than those relying on outsourcing.
- Communication gaps and organisational silos in outsourced setups hinder efficient test integration.
- Resistance to automation adoption is linked to unclear ownership of testing strategies and lack of dedicated automation teams.

This demonstrates that test automation is as much a *social and organisational challenge* as it is a technical one.

Furthermore, the transition to automated testing requires shifts in skillsets, job roles, and decision-making structures. In companies where *test automation is resisted*, we observed a lack of dedicated automation teams or clear ownership of test strategies. This highlights the need for *structured organisational change* to facilitate the adoption of automated testing, ensuring that human and technical systems evolve together.

While this study has framed software testing and automation within a socio-technical perspective, there remains an opportunity to more explicitly apply the conceptual apparatus of socio-technical systems theory to interpret the findings. For instance, Mumford's ETHICS methodology [9] emphasises the participatory design of socio-technical systems, which could be used to interpret the varying degrees of organisational resistance observed in certain firms. Similarly, Trist's original conception of joint optimisation, the alignment of social and technical subsystems [8] offers a useful heuristic for understanding why in-house setups tend to exhibit higher automation maturity. These theoretical tools enable a richer interpretation of our findings beyond descriptive classification, opening avenues for theory-informed interventions in manufacturing contexts.

5.5. Trends and Patterns

To summarise our findings and highlight the interplay between technical, organisational, and human dimensions in shaping test automation outcomes, we propose the conceptual model illustrated in Fig. 4. This model reflects the socio-technical alignment required for effective test automation, emphasising that successful adoption cannot be achieved through technical capability alone, but must also address organisational structures and human collaboration.

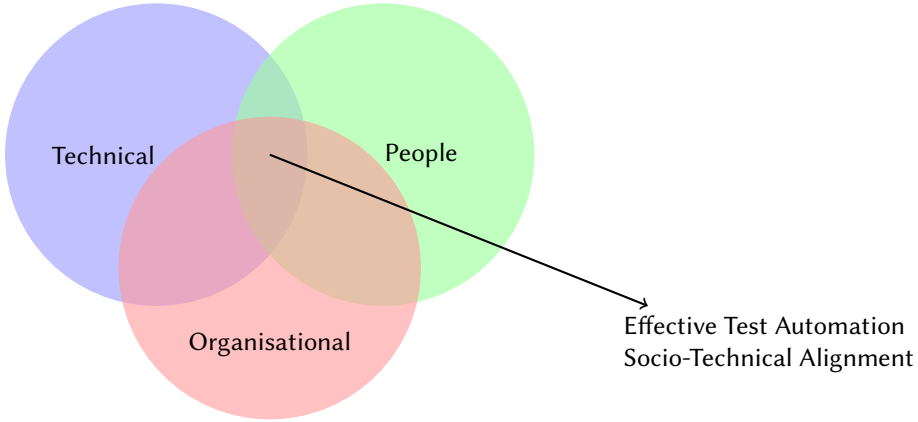

Figure 4: Socio-Technical Alignment for Effective Test Automation

Several notable trends emerged across companies:

- Fully outsourced companies prioritise cost efficiency but lack strong quality assurance practices.
- Manufacturing firms transitioning to digital products often rely on external partners before establishing in-house development.
- Companies adhering to strict industry regulations focus on compliance-driven testing rather than continuous quality improvement.

- Organisations fostering cross-functional collaboration tend to adopt higher levels of automation.

Companies with a full outsourcing strategy have little focus on software testing and test automation. This can be due to the immaturity of the manufacturing company concerning software development. Traditional manufacturing companies that are digitalizing their products might opt out to outsource software development initially to do a large-scale proofing of a concept. This type of software product can even be considered to be market-ready by some manufacturing companies. Still, in reality, it does not meet the quality standards due to a lack of quality assurance performed during development.

On the other hand, mature companies have a very well-established software testing strategy and tend to have more software test automation. Various industry standards and regulations drive some of the larger, mature companies. On one hand, this is deemed beneficial for the quality of the software delivered. On the other hand, this is also perceived limiting to the extent of quality of the software delivered. Mature companies driven by industry standards and regulations tend to limit their quality assurance processes to those alone, and are often satisfied once they are met. This indirectly limits the potential of further development on quality assurance and software testing.

While technical capabilities play a role in determining a company's test automation maturity, socio-technical factors—such as team structures, collaboration between software and hardware engineers, and executive support—are equally important. Companies with strong cross-functional collaboration and dedicated test automation teams report *higher adoption of automated testing*, while those with fragmented software and product development processes struggle with integration.

This aligns with socio-technical systems theory, which emphasises the importance of aligning human and technical components for optimal system performance [9]. Future initiatives in test automation should therefore consider both technical and organisational factors, ensuring that software testing strategies align with existing work practices and team dynamics.

5.5.1. Development Approach

The patterns identified from the interviews connected to the development approaches show that companies which develop software in-house have a better overview of testing and test automation. Hybrid approaches where development of either software or firmware are outsourced give a balanced approach to testing. Fully outsourced projects tend to have very little focus on software testing in the traditional format. We identified the following patterns:

- In-house development is preferred by larger companies (Alpha, Gamma, Theta) and offers greater control over integration and testing.
- Outsourced development is adopted by companies with very large employee bases (Epsilon) or those focusing on cost-efficiency (Zeta).
- Hybrid approaches are used by companies balancing internal expertise with external capabilities (Beta, Eta).

5.5.2. Testing Strategies

When it comes to testing strategies, our analysis shows that companies which conduct software development in-house usually tend to do more software testing in general. We summarise the findings about test strategies below:

- Early testing is common in companies with in-house development (Alpha, Beta, Gamma, Theta).
- Hardware-driven testing is observed in companies where hardware plays a critical role (Beta, Theta).
- Outsourced testing aligns with outsourced development models (Epsilon, Zeta).
- Companies transitioning to new development models (Gamma) or focusing on the physical product (Eta) show varied testing strategies.

5.5.3. Extent of Test Automation

Lastly, we present the findings on the extent of test automation within manufacturing companies. Our findings imply that the company's maturity strongly reflects the extent of software test automation.

- Higher levels of automation are found in companies with in-house or extensive simultaneous development (Alpha, Beta, Theta).
- Medium levels of automation are typical in traditional manufacturing companies with a strong organisational focus on the physical product (Gamma, Eta).
- Lower levels of automation are seen in companies relying on outsourced development and testing (Epsilon, Zeta).

Furthermore, referring back to the related work presented in Section 2, we can see that previous research looks at different techniques which can be used to mitigate challenges regarding software testing and test automation. Here, academia and state-of-the-art research ought to be beneficial for manufacturing companies. Collaboration between the two is crucial both for academia, being able to place and test their findings, but also for manufacturing companies to tackle the challenges associated with software development and software test automation.

5.6. Limitations and other considerations

This study acknowledges several limitations that may influence the generalizability of the findings. Firstly, no standardised definitions of "software testing" and "software test automation" were established prior to conducting the interviews. The interviewees reflected on their understanding of software testing and software test automation, which ultimately proved to be in line with the definitions described in Section 2. Additionally, the term "test" itself was a point of contention among certain participants. Some individuals argued that "test" is an inadequate term, preferring "quality assurance" to describe their activities. Yet again, this shows that companies have different views over the definitions and that no unified definition of test/quality assurance exists. This indicates the need for standardised definitions and terminology to enhance the reliability and validity of our findings. Finally, the number and type of interviewees for this study could provide findings that are different from those that would be obtained if we were to interview a larger number of software developers working in these companies. This is potentially something another study could explore, and conclusions could be drawn based on the combined findings, further discussed in Section 6.

6. Conclusion

This study analyses the current state of software testing and test automation in the manufacturing industry, highlighting key trends and challenges. The findings underscore the diversity in approaches adopted by manufacturing companies, influenced by their internal capabilities, maturity, external partnerships, and organisational characteristics. As demonstrated in Section 5, we see that manufacturing companies adopt diverse strategies for software development and software testing.

A key insight from this study is that software testing and test automation are not merely technical challenges but inherently *socio-technical* endeavours. The success of automation depends not only on tool selection and process integration but also on organisational culture, collaboration structures, and the ability to align testing with product development cycles. Companies that foster cross-functional collaboration and knowledge-sharing between software, hardware, and quality assurance teams are more likely to achieve effective test automation.

However, significant socio-technical barriers remain. Resistance to automation, lack of dedicated testing roles, and fragmented communication between development teams can hinder adoption. Our results suggest that companies transitioning from traditional manufacturing to software-intensive product development need structured frameworks that address both technical and organisational

challenges. These robust frameworks and methodologies ought to bridge the gap between academic advancements in test automation and their practical application in industrial settings.

The impact of automated testing on the overall product development cycle is evident in several dimensions. Companies with higher levels of test automation report shorter lead times and improved product quality, as automated tests facilitate early defect detection and continuous integration practices. However, the full potential of test automation is often unrealised in companies with less mature software development processes or those heavily reliant on outsourcing.

While our findings confirm several known challenges, such as the limited testing maturity in outsourced environments and the benefits of in-house integration, this persistence itself is noteworthy. It underscores the need for a deeper understanding of the social and organisational forces that hinder technological advancement, even when the tools and methods are well understood. By viewing these frictions through a socio-technical lens, we shift the focus from merely identifying gaps to interrogating why these gaps endure.

Future research should explore socio-technical adaptation strategies for industrial test automation. Understanding how companies can optimise automation by aligning technological capabilities with organisational structures and human factors is essential for advancing the field of software quality assurance in manufacturing contexts. By integrating socio-technical perspectives into test automation strategies, companies can create sustainable and scalable testing solutions that enhance efficiency and product reliability.

6.1. Future Research

This study has provided insights into software testing and test automation in Swedish manufacturing companies. However, several knowledge gaps remain that require further exploration, particularly regarding the interplay between technical and social factors in test automation adoption. While prior studies suggest potential benefits [2], empirical validation is needed to determine how such models can effectively align software testing with product development cycles. Secondly, exploring what impact organisational culture and cross-functional collaboration have on test automation adoption and strategies. Finally, there is a need for longitudinal studies to track how companies evolve in their test automation practices.

By addressing these areas, future research can help organisations develop test automation strategies that are both technically effective and socially sustainable.

Acknowledgments

We deeply thank the companies participating in this study, providing essential insights and valuable time. We would also like to sincerely thank the experts whose consultancy and guidance significantly shaped this research's conceptual framework and methodology. Special appreciation goes to the collaborating research company, which provided me with deeper insights into the manufacturing world.

Declaration on Generative AI

During the preparation of this work, the author(s) used Grammarly in order to: Grammar and spelling check. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] I. Sommerville, Software Engineering, tenth ed., Pearson, 2016. URL: https://www.pearson.com/us/higher-education/program/PGM35255.html.

[2] R. G. Cooper, Stage-gate systems: A new tool for managing new products, Business Horizons 33 (1990) 44–54.

[3] K. Beck, et al., Manifesto for agile software development (2001). URL: https://agilemanifesto.org/.

[4] G. J. Myers, C. Sandler, T. Badgett, The art of software testing, 3rd ed ed., John Wiley & Sons, Hoboken and N.J, 2012.

[5] R. van Megen, D. B. Meyerhoff, Costs and benefits of early defect detection: experiences from developing client server and host applications, Software Quality Journal 4 (1995) 247–256. URL: https://doi.org/10.1007/BF00402646. doi:10.1007/BF00402646.

[6] D. M. Rafi, K. R. K. Moses, K. Petersen, M. V. Mäntylä, Benefits and limitations of automated software testing: Systematic literature review and practitioner survey, in: 2012 7th International Workshop on Automation of Software Test (AST), 2012, pp. 36–42. doi:10.1109/IWAST.2012.6228988.

[7] T. J. Y. James, K. Otto, K. Wood, A comparison of design decisions made early and late in development, ICED (2017).

[8] E. L. Trist, H. Murray, Beyond the individual: The psychology of social understanding, Psychology Press, 1993.

[9] E. Mumford, Redesigning Human Systems, Idea Group Publishing, 2003.

[10] G. Baxter, I. Sommerville, Socio-technical systems: From design methods to systems engineering, Interacting with Computers 23 (2011) 4–17.

[11] McKinsey, Digital twins: The key to smart product development — mckinsey.com, https://www.mckinsey.com/industries/industrials-and-electronics/our-insights/digital-twins-the-key-to-smart-product-development, 2023. [Accessed 07-06-2024].

[12] PricewaterhouseCoopers, Digital Product Development 2025 — pwc.ch, https://www.pwc.ch/en/insights/digital/digital-product-development-2025.html, 2024. [Accessed 07-06-2024].

[13] A. F. Sommer, A. Slavensky, V. T. Nguyen, K. Steger-Jensen, I. Dukovska-Popovska, Scrum integration in stage-gate models for collaborative product development — a case study of three industrial manufacturers, 2014, pp. 1278–1282. doi:10.1109/IEEM.2013.6962616.

[14] U. Eklund, J. Bosch, Applying agile development in mass-produced embedded systems, in: Agile Processes in Software Engineering and Extreme Programming, volume 111 of *LNBIP*, Springer, Berlin, Heidelberg, 2012, pp. 31–46. doi:https://doi.org/10.1007/978-3-642-30350-0_3.

[15] R. G. Cooper, A. F. Sommer, The agile–stage-gate hybrid model: A promising new approach and a new research opportunity, Journal of Product Innovation Management 33 (2016) 513–526. doi:10.1111/jpim.12314.

[16] T. Žužek, J. Kušar, L. Rihar, T. Berlec, Agile-Concurrent hybrid: A framework for concurrent product development using scrum, Concurr. Eng. Res. Appl. 28 (2020) 255–264.

[17] F. Kitsios, M. Kamariotou, Stage-gate and agile manufacturing in new product development: A state-of-the art, volume 2020-September, Academic Conferences and Publishing International Limited, 2020, pp. 330–337. doi:10.34190/EIE.20.147.

[18] V. Garousi, J. Zhi, A survey of software testing practices in canada, Journal of Systems and Software 86 (2013) 1354–1376. doi:10.1016/j.jss.2012.12.051.

[19] J. Itkonen, M. V. Mäntylä, C. Lassenius, The role of the tester's knowledge in exploratory software testing, IEEE Transactions on Software Engineering 39 (2013) 707–724. doi:10.1109/TSE.2012.55.

[20] B. Marculescu, R. Feldt, R. Torkar, S. Poulding, An initial industrial evaluation of interactive search-based testing for embedded software, Applied Soft Computing 29 (2015) 26–39. doi:10.1016/j.asoc.2014.12.025.

[21] W. Afzal, S. Alone, K. Glocksien, R. Torkar, Software test process improvement approaches: A systematic literature review and an industrial case study, Journal of Systems and Software 111 (2016) 1–33. doi:10.1016/j.jss.2015.08.048.

[22] V. Garousi, M. V. Mäntylä, When and what to automate in software testing? a multi-vocal literature

review, 2016. doi:10.1016/j.infsof.2016.04.015.
[23] V. Garousi, M. V. Mäntylä, A systematic literature review of literature reviews in software testing, 2016. doi:10.1016/j.infsof.2016.09.002.
[24] V. Garousi, M. Felderer, Worlds apart industrial and academic focus areas in software testing, 2017. URL: www.sigsoft.org/impact.html. doi:10.1109/MS.2017.3641116.
[25] V. Garousi, M. M. Eskandar, K. Herkiloğlu, Industry–academia collaborations in software testing: experience and success stories from canada and turkey, Software Quality Journal 25 (2017) 1091–1143. doi:10.1007/s11219-016-9319-5.
[26] N. bin Ali, E. Engström, M. Taromirad, M. R. Mousavi, N. M. Minhas, D. Helgesson, S. Kunze, M. Varshosaz, On the search for industry-relevant regression testing research, Empirical Software Engineering 24 (2019) 2020–2055. doi:10.1007/s10664-018-9670-1.
[27] W. Lam, P. Godefroid, S. Nath, A. Santhiar, S. Thummalapenta, Root causing flaky tests in a large-scale industrial setting, Association for Computing Machinery, Inc, 2019, pp. 204–215. doi:10.1145/3293882.3330570.
[28] V. Garousi, M. Felderer, M. Kuhrmann, K. Herkiloğlu, S. Eldh, Exploring the industry's challenges in software testing: An empirical study, Journal of Software: Evolution and Process 32 (2020). doi:10.1002/smr.2251.
[29] F. Gurcan, G. G. M. Dalveren, N. E. Cagiltay, D. Roman, A. Soylu, Evolution of software testing strategies and trends: Semantic content analysis of software research corpus of the last 40 years, IEEE Access 10 (2022) 106093–106109. doi:10.1109/ACCESS.2022.3211949.
[30] L. Williams, G. Kudrjavets, N. Nagappan, On the effectiveness of unit test automation at microsoft, in: 2009 20th International Symposium on Software Reliability Engineering, 2009, pp. 81–89. doi:10.1109/ISSRE.2009.32.
[31] G. Fraser, A. Arcuri, A large-scale evaluation of automated unit test generation using evosuite, ACM Transactions on Software Engineering and Methodology 24 (2014). doi:10.1145/2685612.
[32] A. Arcuri, Restful api automated test case generation with evomaster, ACM Transactions on Software Engineering and Methodology 28 (2019) 1–37. doi:10.1145/3293455.
[33] M. Brunetto, G. Denaro, L. Mariani, M. Pezzè, On introducing automatic test case generation in practice: A success story and lessons learned, Journal of Systems and Software 176 (2021). doi:10.1016/j.jss.2021.110933.
[34] S. Bardin, N. Kosmatov, M. Marcozzi, M. Delahaye, Specify and measure, cover and reveal: A unified framework for automated test generation, Science of Computer Programming 207 (2021). doi:10.1016/j.scico.2021.102641.
[35] Y. Wang, M. V. Mäntylä, Z. Liu, J. Markkula, Test automation maturity improves product quality—quantitative study of open source projects using continuous integration, Journal of Systems and Software 188 (2022). doi:10.1016/j.jss.2022.111259.
[36] L. Yu, E. Alégroth, P. Chatzipetrou, T. Gorschek, Automated nfr testing in continuous integration environments: a multi-case study of nordic companies, Empirical Software Engineering 28 (2023). doi:10.1007/s10664-023-10356-1.
[37] V. Siqueira, B. Miranda, An industrial experience report on the adoption of history-based test case prioritization, Association for Computing Machinery, 2023, pp. 110–112. doi:10.1145/3624032.3624048.
[38] M. L. Mohd-Shafie, W. M. N. W. Kadir, H. Lichter, M. Khatibsyarbini, M. A. Isa, Model-based test case generation and prioritization: a systematic literature review, Software and Systems Modeling 21 (2022) 717–753. doi:10.1007/s10270-021-00924-8.
[39] A. Trudova, M. Dolezel, Artificial intelligence in software test automation: A systematic literature review, 2020. URL: http://orcid.org/0000-0002-5963-5145] andAlenaBuchalcevova1[0000-0002-8185-5208].
[40] M. Eckhart, K. Meixner, D. Winkler, A. Ekelhart, Securing the testing process for industrial automation software, Computers and Security 85 (2019) 156–180. doi:10.1016/j.cose.2019.04.016.
[41] R. Feldt, L. Angelis, R. Torkar, M. Samuelsson, Software development waste, Journal of Systems

and Software 144 (2018) 21–43.
[42] T. Majumdar, K. Petersen, M. Mattsson, Cross-functional collaboration in software-intensive product development: A systematic mapping study, Journal of Systems and Software 187 (2022) 111256.
[43] C. Robson, Real World Research, 3 ed., John Wiley & Sons, Chichester, England, 2011.
[44] D. Tilson, K. Lyytinen, C. Sørensen, Digital infrastructures: The missing is research agenda, Information Systems Research 21 (2010) 748–759.
[45] C. A. O'Reilly, M. L. Tushman, Organizational ambidexterity: Past, present, and future, Academy of Management Perspectives 27 (2013) 324–338.
[46] T. Kretschmer, P. Khashabi, Complementarities in organizational design and firm performance: A review and future research directions, Academy of Management Annals 14 (2020) 275–298.
[47] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, A. Wesslén, Experimentation in Software Engineering, Springer Berlin Heidelberg, 2012. URL: http://dx.doi.org/10.1007/978-3-642-29044-2. doi:10.1007/978-3-642-29044-2.
[48] D. Karlström, P. Runeson, Stage-gate project management models, 2005. doi:10.1109/MS.2005.59.
[49] H. Rubin, I. Rubin, Qualitative Interviewing (2nd ed.): The Art of Hearing Data, SAGE Publications, Inc., 2005. URL: http://dx.doi.org/10.4135/9781452226651. doi:10.4135/9781452226651.
[50] S. E. Hove, B. Anda, Experiences from conducting semi-structured interviews in empirical software engineering research, 2005.
[51] C. B. Seaman, Qualitative methods in empirical studies of software engineering, 1999.
[52] P. Lenberg, R. Feldt, L. Wallgren, I. Tidefors, D. Graziotin, Behavioral software engineering - guidelines for qualitative studies (2017). doi:10.48550/arXiv.1712.08341.
[53] D. S. Cruzes, T. Dybå, Synthesizing evidence in software engineering research, in: ESEM 2010 - Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, 2010. doi:10.1145/1852786.1852788.
[54] V. Braun, V. Clarke, Using thematic analysis in psychology, Qualitative Research in Psychology 3 (2006) 77–101. doi:10.1191/1478088706qp063oa.
[55] B. Glaser, A. Strauss, The Discovery of Grounded Theory: Strategies for Qualitative Research, Observations (Chicago, Ill.), Aldine, 1967. URL: https://books.google.se/books?id=oUxEAQAAIAAJ.
[56] M. Wiesche, M. Jurisch, P. Yetton, H. Krcmar, Grounded theory methodology in information systems research, MIS Quarterly 41 (2017) 685–701. doi:10.25300/MISQ/2017/41.3.02.