Peetu Salonen

Implementing Reliability Testing in Medical Device Development

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Bachelor's Thesis

26 October 2024

Abstract

Author:	Peetu Salonen
Title:	Implementing Reliability Testing in Medical Device Development
Number of Pages:	54 pages
Date:	26 October 2024
Degree:	Bachelor of Engineering
Degree Programme:	Information Technology
Professional Major:	Medical Technology
Supervisors:	Markku Ursin, SW Engineering Manager Sakari Lukkarinen, Principal Lecturer

---

In the evolving field of medical device development, it is essential to ensure the reliability and safety of products. This thesis presents a reliability testing process tailored for medical devices, integrating agile methodologies and Continuous Integration and Continuous Delivery (CI/CD) practices. The study examines the importance of early integration of testing and risk-based approaches, highlighting the challenges in regulated environments.

A literature review explores current trends in medical device reliability testing, emphasizing the need for efficient testing frameworks. The software testing standard – ISO 29119 – is examined to identify the industry best practices.

After best practices from literature review and testing standards are identified, the reliability testing process is detailed. It includes planning, development and execution stages. Key methodologies include the use of automation and tools such as Robot Framework and Python for keyword-driven testing, Docker for consistent test environments, and Jenkins and Grafana for continuous monitoring and real-time data visualization. The process focuses on iterative approach with continuous adjustments and modular documentation strategies to reduce overhead while maintaining regulatory compliance. The examples demonstrate how modern reliability testing process is effectively implemented into medical device development in modern fast-paced development environments, helping to ensure product safety and compliance.

Keywords: Medical devices, reliability testing, test automation, keyword-driven testing, medical device safety

---

Contents

List of Abbreviations

PnR: Performance and reliability testing.

MDR: European Union’s regulation on medical devices 2017/745, came into full effect in 2021.

CAPA: Corrective and Preventive Action. MDR requires identifying, investigating, and rectifying product or process related deficiencies to prevent their occurrence and ensuring continuous improvement.

CI/CD: Continuous Integration and Continuous Delivery/Deployment, software development practice where code changes are automatically tested and deployed.

KDT: Keyword-driven testing, testing framework designed to enhance modularity, reusability, and abstraction in test design.

V&V: Verification and Validation, testing ensuring that product meets specified design requirements and user needs. In the context of medical devices, it is the formal testing needed for regulatory compliance.

1 Introduction

This thesis explores the process of integrating automated reliability testing into the development lifecycle of a medical device, including detailed planning and execution of the tests. It provides a brief literature review to offer insight into the latest developments in reliability testing of medical devices and investigate best practices in software testing from the software testing standard **ISO 29119** [1].

Ensuring product reliability before market release has been a long-standing practice in manufacturing, initially driven by financial incentives [2]. Regulatory requirements have refined this approach into an industry standard for medical device manufacturers. With the **European Union's Medical Device Regulation 2017/745 (MDR)** coming into full effect in May 2021, the requirements for clinical evidence and post-market surveillance demand more rigorous and comprehensive testing processes [3].

To address these challenges, medical device manufacturers must guarantee the reliability of all product versions before market release. Introducing new elements to an already established, well-tested system can be laborious and lead to unexpected behaviour or unforeseen issues. Insufficient testing can jeopardize patient safety and expose manufacturer to significant legal and financial consequences. To meet heightened regulatory requirements while ensuring optimal functionality –and above all, patient safety– medical devices are thoroughly tested during their lifecycle. Consequently, there is an increased need for advanced testing methodologies, such as automated reliability testing, to ensure that new and existing devices meet safety standards.

Automation offers significant possibilities for repetitive reliability testing, ensuring the safety and robustness of products in actual use. Automated tests can execute thousands of test cycles – far more than could ever be achieved with manual testing, and beyond what the product is likely to encounter. This level of testing helps manufacturers to have confidence in their products throughout their lifecycle, resulting in product longevity, reduced recall rate and

improved user satisfaction. Focused reliability testing on new features can provide significant value by enabling the development teams to detect and fix issues early. Proven functionality over numerous repetitions under varying conditions further ensures the safe integration of the tested feature into the system.

The process exploration here is based on a real-world application targeting a new feature in a patient monitoring system, aiming to ensure consistent functionality over repetitive use. As the study's work was commissioned by a medical device manufacturer specializing in patient monitoring systems and software, details within the thesis have been generalized due to confidentiality agreements concerning unreleased features or products. This generalization does not compromise the applicability of the methodologies and information, which are grounded on accepted engineering principles and align well with recognized international standards.

The goal of this thesis was to investigate the methodologies, tools and techniques required to create a reliability testing framework suited to the needs of **Continuous Integration and Continuous Delivery (CI/CD)** environments [4]. By implementing this framework, the aim was to streamline the testing process and enhance the overall reliability of medical devices. The framework incorporates tools such as **Robot Framework** for keyword-driven testing, **Python** for scripting, **Docker** for maintaining isolated test environments and **Jenkins** for automating test execution and result tracking [5, 6, 7, 8]. **GitLab** is used as the version control tool, and **Grafana** offers real-time monitoring of metrics, enhancing visibility and communication within the process [9, 10]. Based on real-world applications and reviewed by industry professionals, the process can broadly be applied across similar contexts within the industry. Documented test results can be leveraged to support the validation of features, confirming regulatory compliance and safety for integration into the existing systems.

The structure of the thesis is as follows: Chapter 2 provides an overview of patient monitors, medical device testing and their regulatory requirements.

Chapter 3 presents a literature review on medical device reliability testing and the exploration of ISO 29119 [1]. Chapter 4 explores the process of automated software reliability testing. The final chapter concludes the findings and suggests areas for future development and improvement of the testing process. Figure 1 illustrates the structure for clarity.

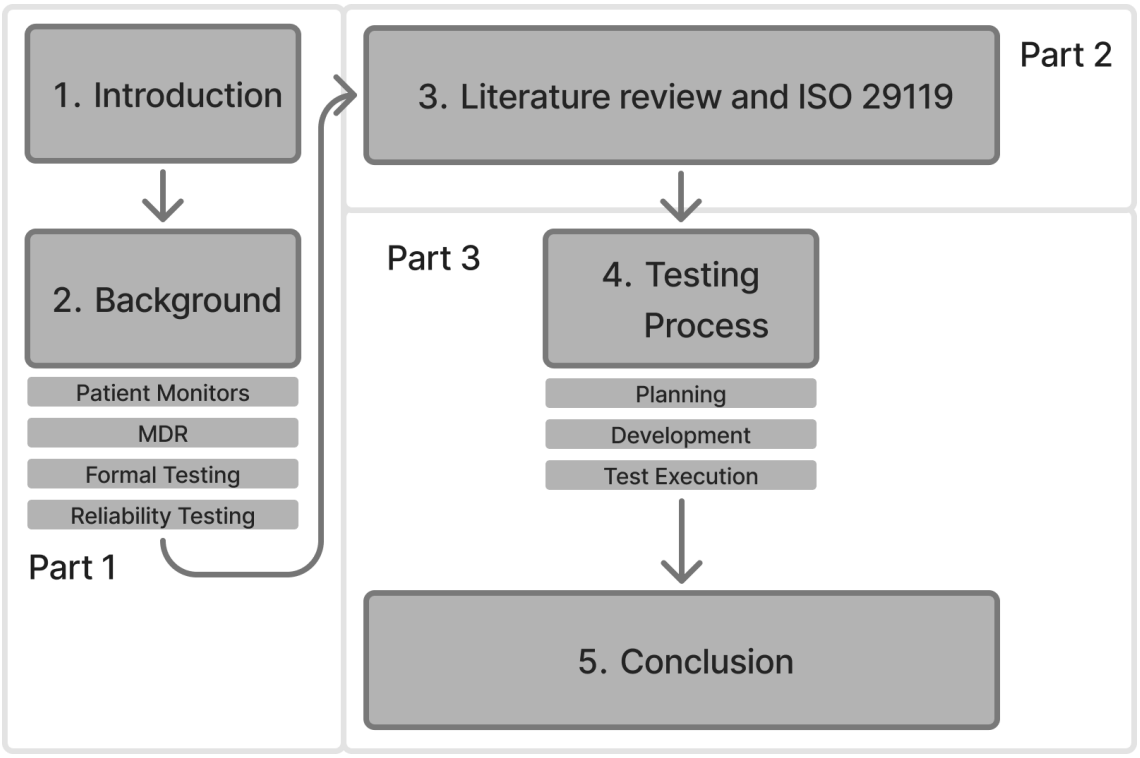

Figure 1. Structure of the thesis.

The text is divided into several key sections: the first covers the background, the second has the literature review and examines ISO 29119, the third details the reliability testing process, ending to the conclusion.

2 Background

2.1 Patient Monitoring Systems

Patient monitors are essential medical devices that play a critical role in modern healthcare by continuously measuring and displaying physiological data from patients to healthcare personnel in real time. This section explores the

functionalities and complexities of patient monitors, discuss importance of data integrity in patient monitoring and how the testing process applies to them.

2.1.1 Overview of Patient Monitors

Patient monitors track a variety of parameters, including:

- **Heart rate via electrocardiogram** (ECG): to monitor cardiac function.
- **Blood pressure** through invasive and non-invasive methods: to assess cardiovascular health.
- **Oxygen saturation and pulse rate**: to ensure adequate oxygen levels in the blood.
- **Body temperature**: to detect fever or hypothermia.

The complexity of patient monitors ranges from simple single-parameter models to advanced multi-parameter systems, depending on the need of the patient case. These advanced systems are crucial in challenging care environments such as **intensive care units** (ICU), **operation rooms** (OR), and during **patient transports**, where clinicians must rely on the accuracy of data to make immediate medical decisions. [11]

Figure 2 offers a graphical view of ICU and transport setups, illustrating the differences. In the ICU patient monitoring is done through parameter cables connected to patient data module and the challenge of the setup is in the number of parameters monitored. Patient data module is charging during its connection to monitor.

When the patient is being transferred, the patient data module acts as transport monitor providing continuous monitoring, changing the challenges to the availability and reliability of the data.

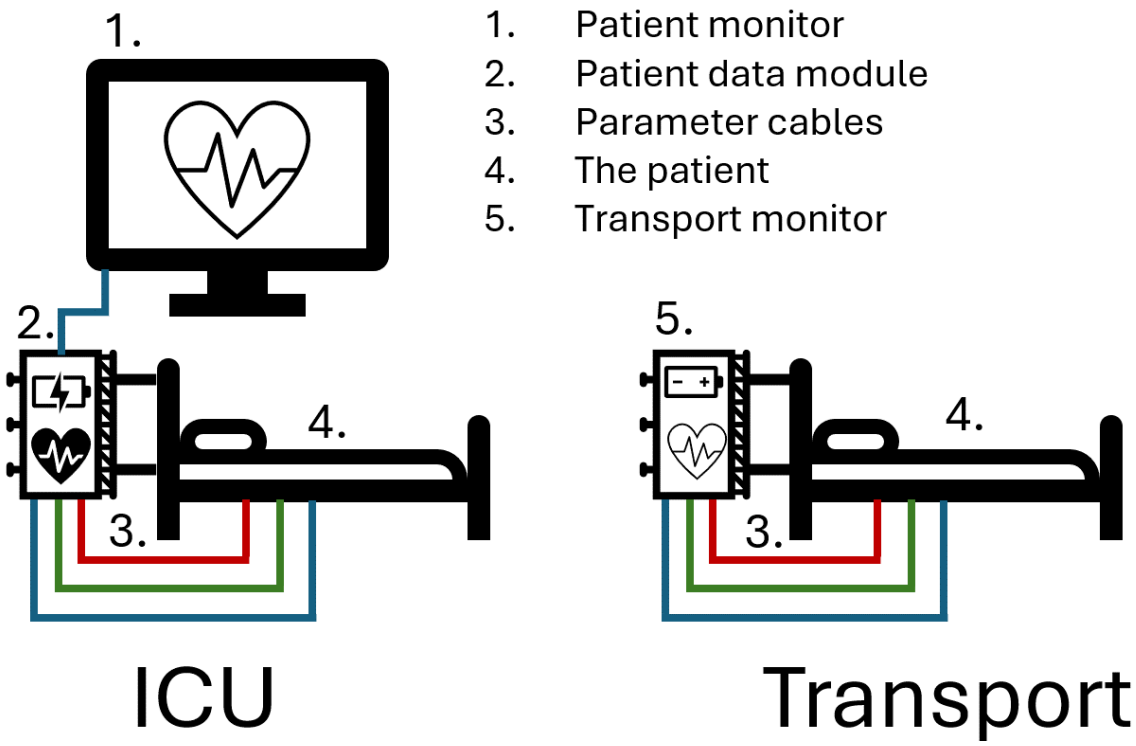

Figure 2. Patient monitoring in ICU and during transport.

Optimal functionality in these demanding setups is vital, and properly executed reliability testing plays a significant role in ensuring it.

2.1.2 Data Integrity in Patient Monitoring

Given the critical importance of real-time data accuracy in patient monitoring, reliability testing is essential to ensure the constant delivery of accurate and dependable data under various real-world medical conditions. Patient monitor reliability testing, such as discussed in this thesis, can target single feature or component of the system but is performed at the system level [1].

2.1.3 System-Level Testing for Patient Monitoring

System-level testing evaluates the feature within the broader system, including all interfacing components such as measurement modules, data cables, and sensors. The testing process simulates the challenging patient care scenarios by including various measurement modules, multiple parameters, and

simulators connected to the device under test. By reflecting real-world conditions, this approach increases complexity and expands testing possibilities, allowing for a more comprehensive assessment of the system reliability.

2.1.4 Application of Testing Process

The process described in here originally targeted a new patient transport feature on a patient monitoring system, enabling continuous monitoring and data flow during transport scenarios with remote monitoring capabilities. However, the framework is applicable to other test development processes, especially in Continuous Integration and Continuous Delivery (CI/CD) environments [4].

2.2 Medical Device Regulation

The development and deployment of medical devices are governed by regulatory frameworks designed to ensure patient safety and product effectiveness. This section delves into regulatory landscape that medical device manufacturers must navigate, introduce industry standard, and emphasize the relevance to our reliability testing process.

2.2.1 Manufacturer Responsibilities

Medical device manufacturers must define the intended use for their products. Under the **European Union's Medical Device Regulation 2017/745 (MDR)** [3]. The intended use is specified through labelling, instructions, and promotional materials. To fulfil these intended purposes while ensuring patient and user safety, medical devices operate within a highly regulated market that requires extensive testing, including safety, performance, and compliance evaluations.

Before products can be released to the market, manufacturers must demonstrate compliance with these regulations by providing objective evidence

that the medical device functions safely and effectively within its intended use. In addition to regulatory approval, robust testing protocols ensure the long-term reliability of the medical devices, further enhancing patient safety and the quality of care. These regulatory requirements are closely tied to the reliability testing process we discuss in this thesis, as it ensures safety and effectiveness within the intended use.

2.2.2 Regulatory Compliance and Industry Standards

Despite the availability of supporting documents and instructions, medical device manufacturers have faced significant challenges since the full implementation of MDR in 2021 [12, 13]. The increased need for clinical evidence and post-market surveillance adds complexity to the regulatory compliance. To assist with these challenges, industry specific standards have been defined to provide guidance. **ISO 14971**, the medical device risk management standard, outlines processes to identifying, evaluating, and mitigating risks throughout product's lifecycle [14]. **ISO 13485**, the medical device quality management system standard, specifies requirements for a quality management system that consistently meets customer and regulatory requirements [15]. Both have technical records acting as user manuals, aiding manufacturers in achieving compliance [16, 17].

ISO14971 provides valuable insights for testing by identifying weaknesses or potential issues within the system, aligning well with the risk-based testing approach we adopt in this thesis. Quality management systems defined in ISO 13485 emphasize continuous improvement, supporting future testing efforts. Our process described in this thesis adheres to these industry standards and is well-suited for environments with ISO 13485 quality management systems.

2.2.3 Relevance to Reliability Testing

Table 1 below summarizes key MDR requirements that our reliability testing process can help to address.

Table 1. Essential requirements of the MDR and how the thesis process contributes to compliance [3].

Requirement	Description	How Our Process Considers It
Risk Management	Manufacturers must have a comprehensive risk management system, such as ISO 14791, in place throughout the entire device lifecycle [14].	The thesis process test objectives are defined by risk-based approaches of the risk management system to identify and mitigate potential failure modes.
Performance Evaluation	Devices must meet safety and performance standards.	Reliability testing simulates real-world conditions, helping to ensure device meets the safety requirements of MDR.
Usability	Medical devices must be designed for safe and effective use, considering the user needs.	The thesis process supports usability by validating device's consistent reliability, directly contributing to safe user operation.
Post-Market Surveillance	Manufacturers must continuously monitor and analyse the device performance and safety after it is on the market, implementing corrective actions as needed.	The testing of the process is continuously executed in CI/CD environment after release. Additionally, the process can be used for post-market Corrective and Preventative Action processes, defined further in ISO 13485 [15].
Technical Documentation	Detailed documentation is needed for regulatory approval, including test protocols, risk assessments, and performance evaluations.	The test process provides documentation to meet the requirements of MDR.
Clinical Evaluation	Clinical data must be gathered to prove the device's safety and	The testing indirectly supports clinical evaluation by ensuring

Requirement	Description	How Our Process Considers It
	performance in real-worlds conditions.	reliability of the device, forming the basis for accurate clinical data collection.
Traceability	Tracking of devices throughout their lifecycle, including design, production, and post-market monitoring.	The thesis incorporates traceability in the testing process.

The regulatory landscape highlights the importance of comprehensive reliability testing of medical devices. Clear and repeatable testing processes ensure that effective testing is conducted to current and future versions of products.

2.3 Regulatory Compliance Testing

Verification and Validation (V&V) are critical processes in medical device development lifecycle. They ensure product meets regulatory requirements and user needs. This chapter provides the overview of the verification and validation testing and discusses their role in regulatory compliance. It also explores their role in establishing safety and effectiveness of medical devices, setting the foundation for the focus on reliability testing.

2.3.1 Verification Testing

Verification is an essential component in bringing a medical device to market, ensuring regulatory compliance and meeting user needs. Verification confirms that specified requirements at the software, hardware, and system levels are met, ensuring that functionality is correctly implemented and operates as expected. We conduct verification using formal testing procedures that consist of manual or automated test cases. Verification demands strict documentation with objective evidence to meet regulatory requirements. Tools used in verification testing must be properly calibrated, and both the test execution

steps and results must be recorded and documented to provide the repeatability and traceability required by regulatory standards [3].

2.3.2 Validation Testing

Validation follows once all components of the product are successfully verified. Validation ensures that the device meets user needs in real-world conditions. We conduct validation against the defined intended use and user requirements of the device, using real-world scenarios. Unlike verification, which is usually conducted internally by the manufacturer, validation involves external parties and actual use environments. These processes fulfil regulatory requirements by ensuring specified functionality with the highest safety standards and efficacy. Best practices within verification and validation (V&V) processes for medical devices are further detailed in industry standards, such as ISO 14971 and ISO 13485 [14, 15].

2.3.3 Differences between Verification and Validation

V&V are critical to ensure safety and effectiveness of medical devices, they serve different purposes. Table 3 provides a comparison between verification and validation, highlighting the key differences.

Table 2. Comparison of verification and validation.

	Verification	**Validation**
Purpose	Ensure product meets design specifications.	Ensure product meets user needs and intended use.
Conducted by	Internal teams: engineers, testers.	External parties: users, clinicians.
Environment	Controlled test environments.	Real-world conditions.
Focus	Correct implementation.	Intended use.

	Verification	Validation
Documentation	Detailed documentation with traceability.	User feedback and clinical evaluations.

Verification answers to the question “Are we building the product right?” by ensuring that product meets specified design requirements. The question validation tries to answer is “Are we building the right product?” by ensuring the device fulfils its intended use in the real-world environment to meet user needs and expectations. [1]

2.4 Reliability Testing

Reliability testing is a critical aspect of medical device development, ensuring that devices perform consistently and safely over time. In this chapter we explore the concepts of reliability testing, its role in regulatory compliance, and provide practical examples.

2.4.1 Role in Regulatory Compliance

While the reliability testing discussed in this thesis is not a formal part of Verification and Validation (V&V), it plays a critical supporting role in both processes. By exposing the device to stress-testing scenarios, one supports the capability of the product to withstand real-world usage conditions while operating within specifications, thereby reinforcing both V&V efforts.

Documented results of performance and reliability testing can be leveraged to support validation of the device by incorporating the test plan and execution results into the process. Furthermore, demonstrating continuous improvement is a key component of post-market surveillance required by regulations, where reliability testing serves as an effective tool. By conducting reliability testing, we provide evidence that the device meets user needs consistently and reliably over time, further supporting the overall safety and efficacy required by the regulatory standards.

Performance and Reliability testing (PnR) can be used as a tool in post-market Corrective and Preventive Action (CAPA) process, which is triggered by issues found in the install base of the device [3]. While traditional corrective actions – such as work instructions, specifications, and verification – address immediate issues as a corrective aspect of CAPA, PnR can be utilized as a preventative action. By expanding test coverage with tests designed to find similar issues, we can execute them on every iteration or release candidate of the product. This demonstrates that the issue is not only resolved but that potential issues are being proactively assessed. Both defining new tests and expanding existing testing can benefit from using the reliability testing process described in this thesis.

2.4.2 Performance and Reliability Testing

While regulatory framework sets the baseline testing needs for medical devices, different types of non-regulated tests, such as **Performance and Reliability testing (PnR)**, are crucial for ensuring device robustness and safety of use. PnR allows tester to define the frameworks or technologies used to achieve predefined goals and metrics. It can target a single feature or the entire system. These tests aim to confirm that functionality remains consistent over hundreds and thousands of repetitions, under varied conditions, while monitoring defined performance and reliability metrics.

Performance and reliability testing are often executed together because of their complementary nature. Reliability testing adds stress factors to the system with long repetitive tests over extended run times. This approach provides effective opportunities to measure performance metrics, such as response times and resource usage under load. PnR is especially valuable in identifying issues and addressing potential weaknesses or failure modes. Consistent execution of PnR tests during the development phase, particularly in environments utilizing CI/CD, provides valuable insights. It can spot potential hidden regressions, such as subtle performance declines or increased reboot times with each deployment. These issues can accumulate unnoticed by human user, possibly indicating bigger underlying problems within the system.

2.4.3 Test Examples

Examples of Performance and Reliability testing (PnR) include automated use case testing, power cycle testing, and stress tests targeting user interface (UI) and system connections. Table 3 gives brief explanation of each.

Table 3. Performance and reliability testing examples.

Test Type	Description
Automated Use Case Testing	Simulate real-world scenarios with repetition to place load on the system.
Power Cycle Testing	Test how the system returns to operational state after power cut or unexpected reboot.
UI Stress Testing	Simulate continuous user actions while measuring metrics.
Connection Reliability Testing	Provide artificial disruptions, such as network overload, while testing connection in different situations.

Automated use case tests simulate various device scenarios, placing load on the system through long execution times and repetition. For instance, in a patient monitoring system, a simulated patient case might start from admitting patient into the ward, transport to and from operating room, and transfer to intensive care unit, with simulated stress test events occurring at all these stages.

Power cycle test checks that the system returns to operational state within the time specified after a power cut or unexpected reboot. These tests offer great insight into the state of the system, its resilience and potential regressions, especially when executed regularly during development phase across different deployments.

Stress tests targeting UI involve continuous simulated user actions, while connection stress test can introduce artificial disruptions, such as network traffic

overload or repetitive cable connections and disconnections. These stress tests simultaneously monitor metrics, such as UI, event, or action response times, parameter connect times and network connect reliability. They ensure system reliability, network packet flow, and error-free data transfer during extreme conditions.

In patient monitoring, PnR tests usually involve multiple parameters and simulators with additional stress factors, such as artificial network traffic. These tests are invaluable during the development phase and hold significant value for released products, identifying weaknesses, issues, or improvement opportunities. PnR benefits from a diverse array of tools, such as simulators providing input data to the system or scripts performing actions within the device. Performance and reliability testing runtimes are often very long – for example 10 000+ cycles or 40 days of testing – which necessities test automation.

3 Literature Review and Testing Standards

Before delving into the reliability testing process, it is important to understand both the best testing practices, and the latest trends and key elements of medical device reliability testing. To that end, a literature review was conducted to examine the current landscape of medical device development and automated reliability testing. After the literature review, the software testing standard ISO 29119 is inspected for practices relevant to the process.

3.1 Literature Review

3.1.1 Introduction

The related publications for the review were identified through a systematic search of academic databases, including IEEE Xplore and PubMed. Keywords such as “reliability testing”, “medical device”, and “test automation” were used in various combinations. The search was limited to results from the last five years to ensure relevance. After a manual review, six articles were selected for

inclusion. This is not an exhaustive systematic literature review of the topic, but rather investigate the key elements in latest publications on the subject.

3.1.2 Individual Study Summaries

An Approach to Reliability Growth for Medical Device Development

In his 2021 study, Rittgers emphasizes the importance of integrating reliability testing early in the medical device development process [18]. Implementing reliability testing late significantly increases workload and complicates identification of critical system components and the selection of appropriate testing methods.

To identify testing needs, Rittgers employed Failure Modes, Effects, and Criticality Analysis (FMECA), reflecting a risk-based approach. He highlights the importance of focused test objectives and quick feedback, utilizing both top-down and bottom-up testing approaches. According to Rittgers, setting reliability goals and test objectives as early as possible in the development phase is highly valuable. Existing installed base (IB) device data was used as a baseline, leveraging existing data where applicable to establish benchmarks.

Early availability of prototypes through Continuous Integration and Continuous Deployment (CI/CD) practices enhances the value of initial testing by enabling defects to be found and retested sooner, aligning with the common break-fix-test approach of the development cycle.

Rittgers introduces the concept of "test trimming", which involves testing partially implemented components, acknowledging challenges of system-level testing early in the development phase. He recognizes that everything can’t be tested and fully implemented reliability testing requires a fully implemented system.

Finally, Rittgers combines target reliability specifications with target confidence levels, employing reliability growth charts to visually represent test coverage and progress providing clear graphical overviews of testing status.

Ensuring Software Quality Through Effective Quality Assurance Testing: Best Practices and Case Studies

In his 2023 study, Amit Bhanushali explores methods for achieving software quality by drawing from multiple publications and case studies [19]. He recognises testing as the primary tool for ensuring software reliability while acknowledging that increasing software complexity complicates testing simultaneously. According to Bhanushali, a lack of reliability testing leads to many issues within software quality.

Bhanushali highlights the importance of usability and user experience in real-world contexts, seeing software quality assessment as a significant issue that can be addressed by clear and repeatable approaches. Furthermore, he has concerns on compatibility, which can be targeted by system level reliability testing. The study advocates for a risk-based approach to save time and focus with efficient testing, stating that the testers to be well-informed of the product, its users and operating platforms.

A key point of the study is that early detection of issues significantly saves time and money, as developers spend substantial time fixing bugs. This workload increases vastly later the issues are found.

Bhanushali emphasizes the importance of clear communication and immediate feedback loops between testers and developers to enhance early reporting and continuous improvement. Additionally, the study supports combining continuous testing and testing in production, acknowledging the needs of agile methodologies and effective communication.

Critical Device Reliability Assessment in Healthcare Services

A 2023 literature review by Abd Rahman et al. took closer look into methodologies and challenges in assessing medical device reliability [20]. The results highlight three main topics: risk management, performance prediction using machine learning, and medical device management systems. The study acknowledges the need for enhanced strategies to improve overall device reliability in high-risk medical equipment.

Medical device reliability is a critical aspect of any healthcare service, ensuring continuous operation and preventing patient harm. The study emphasizes that reducing variability and risk-based approaches are essential to achieve reliability. Assessing medical device reliability can be complex and limited access to healthcare facilities calls for comprehensive testing.

A Case Study of Agile Software Development for Safety-Critical Systems Projects

In 2020 study, Islam and Storer investigate the adoption of agile software development in safety-critical regulated systems [21]. Although conducted withing aerospace company, its findings are highly relevant to medical devices. The study highlights the challenges of implementing agile practices in highly regulated fields.

Regulatory restrictions in safety-critical industries have been hindering the use of agile methodologies. These industries typically have longer release cycles, which contradict the flexible and fast-paced nature of agile approaches. Combining hardware and software development further increase the length of release cycles.

Significant upfront design work and rigid processes can make adaptations difficult. The study notes that code quality tools, especially those built into CI/CD pipelines, are valuable in safety-critical environments and safety-critical industries have more formalized and less frequent interactions with customers.

Continuous Integration and Delivery Practices for Cyber-Physical Systems: An Interview-Based Study

In their 2023 study, Zampetti et al. examine the integration and challenges of Continuous Integration and Continuous Delivery (CI/CD) practices in cyber-physical systems (CPS) [22]. The study was conducted through ten semi-structured interviews, an external survey and case studies. CPS combine physical and digital components, consisting of software that interacts with physical processes through sensors, networks and actuators. They use real-time data to monitor, control and influence physical systems, emphasizing algorithm-based feedback loops, real-time operation and connectivity. Patient monitoring systems are a prime example of such system, relying on real-time data from various peripherals.

Zampetti et al. highlight that testing on CPS is more difficult to design, develop and execute than traditional software systems. The complexity of CPS makes automation harder, sometimes nearly impossible. Hardware restrictions, scalability issues, and costs are significant challenges that lead to the common practice of using simulators instead of real environments. However, simulators cannot account for the variations in real use environments with actual hardware and data, potentially hiding issues. Testing with real components confirms interactions between them and the system, providing more reliable results. Therefore, quality assessment based solely on tests with simulators is insufficient.

To save time and increase continuous deployment, test runs in the pipeline may simulate things faster than possible in reality. Timely defect fixing is a benefit of proper CI/CD pipeline, but quick retries are rarely and option in failure cases due to the complexes involved. Zampetti et al. advocate for careful planning and focus to save resources, emphasizing risk-based approaches and test prioritization. Additionally, the complexity of CPS makes expected testing behaviours, or oracle specifications, difficult.

Methods for Increasing and Assessing Reliability of Medical Equipment

In their 2020 study, researchers from Tashken State Technical University present methods for enhancing and evaluating the reliability of medical equipment, focusing specifically on microcontrollers within devices [23]. Although centred on a specific component, the principles discussed are broadly applicable to medical device reliability testing.

The authors recommend a focus on repetitive stress testing and emphasize the importance of simulating real-world conditions. This approach aims to detect latent faults that may not surface during typical functional testing but could lead to failures in actual use. They underscore the consequences of medical device failures, emphasizing the critical nature of robust reliability testing.

Metrics used in their study include failure rates and probability of uptime, evaluating the ability of the product to maintain operability over different time intervals. These metrics provide quantitative measures of reliability and are essential for evaluating test strategy effectiveness.

3.2 Thematic Analysis

3.2.1 Early Integration and Risk-Based Approaches

In the thematic analysis, it was observed that the literature consistently emphasizes the critical importance of integrating reliability testing early in the development process. There is a consensus on the importance of risk-based approaches in testing. Prioritizing testing efforts based on risk assessments ensures critical areas receive needed attention and resources are allocated optimally. [18, 19, 20, 21, 22, 23]

Rittgers highlights that initiating reliability testing when features are ready, rather than waiting for the entire system to be complete, significantly reduces workload and facilitates the identification of critical system components. He acknowledges that everything cannot be exhaustively assessed and fully implemented reliability testing requires a fully implemented system. Rittgers utilized approach where test goals were set by architects and finer technical

details were addressed through a bottom-up approach. This dual strategy helps the process by ensuring comprehensive coverage while allowing flexibility in addressing specific technical challenges. Leveraging existing IB device data is another strategy advocated by Rittgers. At the same time, challenges with new features lacking historical data are acknowledged, necessitating expert testing techniques, and highlighting the need for adaptable testing methodologies. [18]

Bhanushali echoes the significance of reliability testing as early as possible in the development phase, especially with the complexity of contemporary software leading to complex test scenarios. It is noted that a lack of reliability testing causes unpredictable software system behaviour, reduced scalability, fault tolerance and dependability. Additionally, inadequate testing negatively affects the construction, definition and management of interfaces and communications. To maximize the value of testing within given resources, identifying appropriate test techniques, methods, and tools is to be emphasized, while analysing the current testing to identify improvement opportunities. Well-defined testing processes can address all these issues. [19]

Abd Rahman et al. define medical device reliability as the probability that a device will function over specified time periods without failures. They highlight the interconnected and complex nature of medical devices with limited access to real use environments, healthcare facilities. This further emphasizes the need of high-quality testing practices. Real-world use can introduce unexpected stresses to the devices, where unexpected maintenance and downtime are costly and patient safety is vital priority. [20]

3.2.2 Agile Methodologies in Regulated Environments

In examining the challenges of agile methodologies in regulated environments, Islam and Storer note that adapting to agile methodologies in regulated fields such as medical technology, can be challenging due to conflicting priorities. The combination of long release cycles and inclusion of both hardware and software development, particularly in regulated environments contradict the fast-paced agile methodologies. Aligning iterative software cycles with longer hardware

development timelines while maintaining regulatory compliance is challenging. Both, hardware development and regulatory requirements require agile practices to adapt, while striving to maintain the benefits of flexibility and responsiveness inherent in them. [21]

Agile methodologies aim to minimize documentation, while regulatory requirements mandate extensive documentation. Additionally, the upfront nature of design and difficulty to adapt in regulated fields, can cause issues within agile methodologies. It is recognized that longer release cycles allow for more extensive development phase testing and should be considered in the testing processes. While Islam and Storer acknowledge the difficulties of adapting to agile methodologies within regulatory requirements in the study, specific solutions are not included in detail. [21]

3.2.3 Real-World Testing and Automation Challenges

It is observed that both Abd Rahman et al. and Zampetti et al. recognize the importance of real-world testing conditions and the need to balance the use of simulated and real hardware in testing. Conducting testing in healthcare facility, such as hospital, is often impossible, the need of similar environments and quality data sources for reliability testing is vital. Automating such complex environments poses another challenge making system testing for a patient monitoring challenging to plan, develop and execute. Focus and planning is increasingly important as the system gets more complex. [20, 22]

Long build times can be a reason why the real-environment testing can be separated from the CI/CD pipeline in CPS. Builds are already taking long time to deploy, and added error prone variability and complexity of real hardware poses threats. To address these challenges, Zampetti et al. identify potential solutions such as continuous test runs in the pipeline and periodic test runs using hardware-in-the-loop or the “green build” rule where testing with real hardware is performed after simulated tests pass. [22]

Zampetti et al. recognize that developed custom simulators can alter the testing to benefit the development teams not taking all the details into account thus hiding issues. It is understood that clear, traceable processes and setups make test reliable and repeatable allowing high-quality specifications and testing. Additional challenge in Continuous Integration and Continuous Deployment (CI/CD) environments in within medical device industry is that re-using artifacts is often not possible due to regulatory traceability requirements. [22]

Tashken State Technical University researchers also highlight the importance of testing in real-world conditions, focusing on repetitive stress testing that simulate real-world conditions while monitoring products ability to maintain operability over time and failure rates. They emphasize the dire consequences of medical device failures, which properly executed reliability testing can avoid. [23]

While these studies acknowledge the challenges of automated testing in complex systems and importance of real-world conditions, there is limited practical guidance on overcoming these obstacles.

3.3 Key Takeaways for Reliability Testing Process

To address key points highlighted by the literature review, they are listed into a table (Table 4). The observations are listed and linked to actions taken in the reliability testing process of this study to ensure the use of best practices resulting to high-quality testing.

Table 4. Key takeaways from literature review to consider in reliability testing process.

ID #	Observation	Source	Process Action
1	Risk-based approach.	[18, 19, 20, 21, 22, 23]	As with most medical device testing, test objectives and specifications are created with risk-based approaches. However,

ID #	Observation	Source	Process Action
			as they were designed from top-down, the thesis process will not dive too deep into their details.
2	Early integration of reliability testing.	[18, 19]	The process is started at the initial stages of development while working closely with the development team. Test trimming is performed as needed to keep the testing and its continuous feedback loops going.
3	Reliability – as the probability that a device will function over specified time periods without failures.	[20, 23]	Long execution times and rigid specifications demand many cycles of test execution.
4	Clear and repeatable processes.	[19]	Goal of the whole process, considered with every part.
5	Combining top-down and bottom-up approaches	[18]	Combining them with top-down specifications and bottom-up finer details. Giving clear objectives while allowing flexibility.
6	Importance of Real-world testing and data	[20, 22, 23]	Use of patient data simulators, real connections and hardware devices whenever possible.
7	Clear communication and immediate feedback loops	[19]	Visualization tools and remotely accessible automated environments with results.
8	Rigid processes and adaptation to changes	[21]	Iteration based process, where adaptations are possible throughout the development phase.

ID #	Observation	Source	Process Action
9	Challenges in agile environments	[21, 22]	See #2, #7, #8
10	Automation challenges in complex systems	[20, 22]	Utilizing existing frameworks and modern tools. Working closely with development teams, allowing solutions for automation. See #2
11	Leveraging existing data from install base	[18]	Whenever applicable. In the case of new feature this is limited.

The reviewed literature highlights the importance of properly planned, developed, and executed reliability testing in medical device development, which can be achieved with well-defined processes.

3.4 Software Testing Standard: ISO 29119

To further improve foundation for the reliability testing process, in this section **ISO 29119** – an international standard providing the guidelines for testing practices – is examined to adapt to the recognized best practices in software testing.

3.4.1 Introduction to ISO 29119

ISO 29119, Software and systems engineering – Software testing, is an international standard that defines terminology and best practices in software testing. Developed to unify software testing practices across industries, it provides a structured approach to designing, executing and managing software testing within organizations and for specific software under test.

Initially published in 2013 with its first three parts, ISO 29119 emerged from the need for consistent and high-quality software testing standards, especially as

software systems grew increasingly complex in critical sectors such as aerospace and medical devices. The standard was later expanded with two additional parts focusing on test techniques and keyword-driven testing, reflecting the shift of the industry towards advanced methods and automation.

ISO 29119 consists of five parts [1]:

- ISO 29119-1: General concepts.
- ISO 29119-2: Test process.
- ISO 29119-3: Test documentation.
- ISO 29119-4: Test techniques.
- ISO 29119-5: Keyword-driven testing.

ISO 29119 provides high-level testing concepts and definitions applicable across multiple industries, including medical devices. ISO 29119-1 covers general concepts and definitions, while ISO 29119-2 focuses on test processes, emphasizing repeatability, traceability and documentation.

ISO 29119-3 offers comprehensive guidelines and best practices for software testing documentation. It includes examples and templates covering the entire testing process and device lifecycle, allowing documentation in electronic or paper-based forms.

ISO 29119-4 outlines a broad range of test techniques designed to enhance coverage, effectiveness, and traceability throughout the software testing lifecycle. These techniques are categorized into specification-based, structure-based and experience-based approaches. ISO 29119-5 introduces Keyword-Driven Testing (KDT), enhancing modularity, reusability, and abstraction in test design.

These five parts of the standard aim to provide effective, consistent, and repeatable testing practices applicable to all types of software across various domains, including embedded systems, mobile applications, web software, scientific computing, defense systems, and medical devices. It shares

similarities with regulatory requirements in the medical device industry, particularly in documentation practices and testing principles.

3.4.2 Compliance with ISO 29119

Organizations can claim conformance to any part of ISO 29119 separately, with the level of tailored conformance needing to be described, rationalized, and agreed upon. Requirements for the full compliance, such as separate testing project alongside development project, seem unnecessary for many software companies. In agile environments, development is increasingly test-driven with developers testing each feature they create. The stricter full compliance aims to maximise test independence by separating testers from developers. Modern agile practices align with the dynamic examples provided in the standard, suitable for tailored conformance.

Many companies, especially in the medical device industry, integrate testers and designers within the agile development teams. This integration helps maintain regulatory compliance in fast-paced development environments. Designers ensure specifications remain uncompromised, and testers maintain the traceability, adapting tests in real time. These practices prevent technical debt related to regulatory compliance. Medical device companies often already meet the requirements for traceability and thorough documentation due to regulatory requirements.

The reliability testing process applied here does not focus on the formal verification, allowing flexibility in its documentation while meeting necessary compliance standards. Performance and stress testing remain as essential testing components in all the examples of the ISO 29119.

In the following section, the key elements for reliability testing provided by ISO 29119 are identified to ensure the use of best practices in the testing process. The standard remains highly relevant and is frequently updated, with many organizational testing practices today based on its guidelines.

3.4.3 Key Components for Reliability Testing

Elements relevant to the process in question were identified from ISO 29119.

Table 5 presents them and how the reliability testing process accounts for each.

Table 5. Key components from ISO 29119 to consider for reliability testing process.

Aspect	ISO 29119	Testing Process
Terminology	Uses standardized terms and definitions.	We adopt core concepts but use simplified terminology to enhance efficiency of internal communication.
Risk-Based Approach	Emphasizes a risk-based approach as the foundation of any testing activities.	We adopt risk-based approach as the foundational principle in our reliability testing process, ensuring patient safety and regulatory compliance.
Test Independence	Recommends balance to maintain objectivity within resources.	We achieve balance by having dedicated testers who maintain good communication with developers.
Planning and Traceability	Emphasizes detailed planning with clear definition of test objectives. Requires comprehensive documentation.	We maintain traceability as its essential for regulatory compliance. Our iterative approach allows flexibility while ensuring traceability.
Test Environments	Requires detailed documentation of test setups and configurations to ensure traceability and repeatability.	To address complexities of system-level testing, we define the core components for traceability while allowing the testers flexibility on finer details.
Test Techniques	Recommends combining multiple test techniques.	We use combination of different testing techniques while

Aspect	ISO 29119	Testing Process
		performing functional reliability tests simultaneously with non-functional performance tests.
Adaptability to Agile Environments	Acknowledges agile methodologies but does not fully encompass dynamic CI/CD environments.	We utilize automated continuous monitoring tools, allowing dynamic adjustments and real time tracking for stakeholders.
Test Automation	Encourages automated testing for repetitive actions and emphasizes importance of automated regression tests.	We use automated methods in reliability testing whenever possible. Our process excels in regression with immediate feedback loops.
Keyword-Driven Testing (KDT)	Strong recommendation for KDT. Part 5 is dedicated solely to this.	We use KDT in test automation, improving modularity, repeatability, and test case maintainability.
Real-Time Data and Results	Does not emphasize real-time data visualization tools.	We incorporate real-time visualization for visibility and immediate stakeholder feedback bringing value to CI/CD environment.
Modular Documentation for Agile Compliance	Requires extensive, and rigid documentation for traceability.	We navigate regulatory field by modular documentation strategy, reducing overhead.
Improvement Opportunities	Suggests use of advanced testing techniques and continuous improvement.	We recognize the possibilities of adding automated exploratory testing or statistical analysis to our process.

Each aspect was investigated to provide more of them regarding the process.

Terminology

By the terminology of ISO 29119, the testing process includes various testing types and levels, such as compatibility, load, performance, portability, regression, risk-based, specification-based and stress testing. While adopting the core concepts of ISO 29119, simplified terminology to enhance internal communication and efficiency is used. This practical adaptation ensures mutual understanding within the organization without strictly following the terminology of the standard.

Risk-Based Approach

As highlighted in the literature review, it is highly recommended to use risk-based approaches to testing, especially with medical devices where safety issues directly impact patient safety and regulatory compliance. A risk-based approach was adopted as the foundational principle in the reliability testing process prioritizing testing activities based on potential impacts and failure probabilities. In medical device development regulations demand all residual risks to be mitigated as far as possible, which extends the risk-based approach even beyond the framework of the standard.

Test Independence

Test independence refers to the degree of separation between testers and developers. Higher independence can lead to more objective testing but may require more resources. Conversely, lower independence can save time but might reduce objectivity. Balance is achieved by having dedicated testers developing the tests while maintaining timely communication with the developers. This approach allows effective testing through collaboration without compromising objectivity.

Planning and Traceability

ISO 29119 emphasizes traceability and clearly defined test objectives, including purpose, scope and goal of the tests. Traceability is maintained through unique

identifiers and change histories, necessity for regulatory compliance. ISO 29119 introduces re-planning as a control activity to manage processes iteratively. The iterative approach here adapts to the demands of medical device development within CI/CD environment.

Test Environments

ISO 29119 requires detailed documentation of setups and configurations. In the process a more practical approach was adopted by defining only the essential components, granting the testers flexibility in building and adjusting the setups as needed to meet the test objectives. This approach addresses the complexities of system-level testing in real-world scenarios with dynamic testing methodologies and automated documentation practices.

Test Techniques

ISO 29119 emphasizes the importance of combining test techniques to achieve comprehensive test coverage. In medical device testing, specification-based techniques are crucial for ensuring regulatory compliance, providing objective evidence and traceability. The reliability testing process heavily utilizes different techniques while having test objectives and expected behaviour acts as basis of specifications tested. For example, stress testing is combined with random inputs within the valid input ranges instead of focusing solely on boundary values. This approach is particularly effective for tests with a high number of repetitions, as it increases the likelihood of uncovering unexpected system behaviours that might not appear with fixed or boundary values, enhancing the overall robustness of the testing process. Additionally, by combining performance testing with functional reliability tests, this element is further enhanced.

Adaptability to Agile Environments

Even though ISO 29119 acknowledges iterative review and continuous monitoring, it does not fully encompass the dynamic, real-time adjustments of

the process. Here, automated continuous monitoring tools are used to capture and display metrics and results in real time throughout the development stage. This approach improves the application of the process to fast-paced CI/CD environments and agile methodologies.

Test Automation

ISO 29119 encourages automated testing for any repetitive activities, stating that if the test case is executed more than five times, automation is justified. It recognizes the importance of continuous execution of automated tests between integrations in a modern CI/CD environment, advocating automated regression testing between deliveries. With automated reliability testing, it is possible to improve coverage, reduce manual errors, and enhance repeatability. The process here excels in regression testing, providing real-time results and visual presentations between deliveries.

Keyword-Driven Testing

ISO 29119 strongly recommends **Keyword-Driven Testing (KDT)**, dedicating its last part to this subject. The reliability testing process uses KDT by incorporating automated testing using keyword-driven Robot Framework (RF) and Python libraries. The implementation of KDT improves modularity and test case maintainability by separating test logic from tool-specific commands, making it particularly suited for medical device environments. Additionally, the modularity gained by separating test logic from test data enhances traceability and repeatability, which can be further increased with isolated test environments by using Docker containers.

Real-Time Data and Results

Continuous monitoring and real-time data collection are used to capture metrics and results throughout the testing process. Tools such as Jenkins and Grafana provide immediate visibility and feedback to stakeholders, enhancing communication and reducing delays. The real-time approach is critical in

detecting performance declines or system failures quickly in CI/CD environment.

Modular Documentation for Agile Compliance

ISO 29119 requires extensive documentation for traceability with results stored within the test procedure documentations. The process adopts modular documentation strategy to reduce overhead without compromising the essential information. Test plans across multiple versions of the software are reused by attaching new documents to original plan, preserving its integrity. This approach is specifically useful in agile environments in regulated fields as the documentation changes can be costly and time-consuming.

Improvement Opportunities for the Process

Although it is challenging to automate expert testing techniques, it is recognized that the keyword-driven reliability testing could benefit from automated exploratory testing elements, such as fuzz testing. Additionally, by applying statistical methods to analyse test results over the numerous executions and deliveries, it would be possible to identify patterns or anomalies indicating underlying issues.

4 Reliability Testing Process

In the highly regulated field of medical devices, ensuring that system operates reliably under all expected conditions is essential. Such level of reliability can be achieved through well-defined testing processes. This chapter presents the reliability testing process developed for medical devices. It also discusses the reasons for modifying or introducing reliability tests with practical examples.

4.1 Overview of Reliability Testing Process

Existing reliability tests are typically modified, or new ones are introduced for one of the following critical reasons:

- Introducing a new feature.
- Identifying a gap in current testing.
- Addressing a Corrective and Preventive Action (CAPA) process [3].

New features within a system, whether they are software functionalities or new hardware components such as cable connections, significantly benefit from inclusion in system-level testing. This is especially true in the context of reliability tests, which ensure new additions function correctly as part of the system under extreme conditions over extended run times. New features in medical devices needs to be validated and our reliability testing results can support the validation process.

When an improvement opportunity is identified in the current testing, expanding existing tests or adding new ones is necessary to fulfil the continuous improvement efforts mandated by regulation. Furthermore, while the CAPA process primarily focuses on corrective actions, which can be achieved through official processes, reliability testing can be utilized as a preventative measure, helping to prevent similar issues in the future.

The reliability testing process presented in the present study was initially created for a new feature in an existing system, necessitating completely new tests. However, it is equally applicable for expanding existing test coverage. The process was split into three key stages: **Planning**, **Development** and **Execution**. During the planning phase, the metrics based on the test objectives along with key element of the test setup were defined. The development phase involved creation of the automated test cases, ensuring their focus and efficacy. In the execution phase, test run and result monitoring were conducted while addressing any issues found. Finally, the test results were documented to support with regulatory compliance.

Figure 3 provides a graphical presentation of the reliability testing process, illustrating the journey from initial triggers – such as the addition of a new feature – through the stages of planning and executing tests, and finally ending with the results being stored.

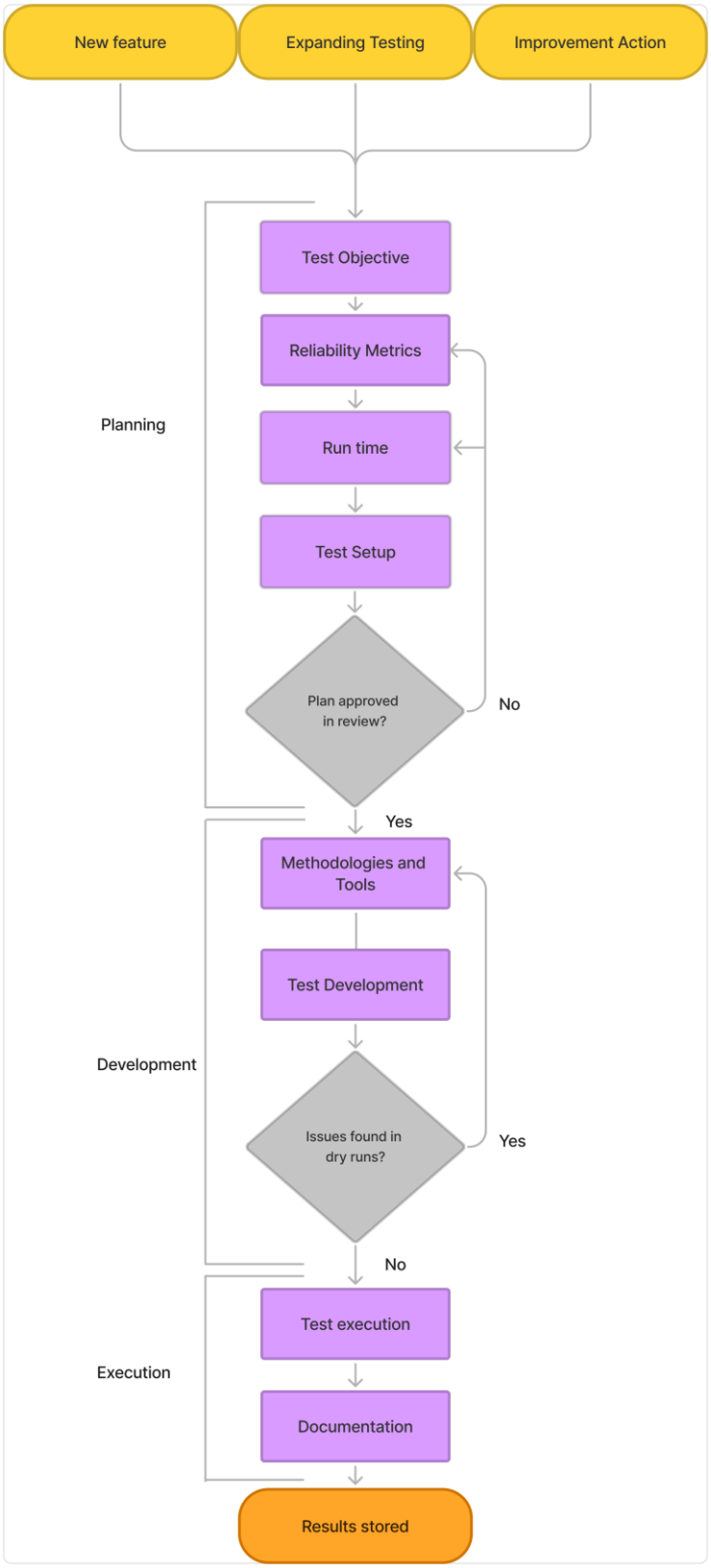

Figure 3. Overview of the reliability testing process.

With understanding of the overall process, the text now provides an in-depth exploration of each stage, including the methodologies and tools used. Due to confidentiality constraint, the real process cannot be explained. Instead, an example test case – **Remote Monitoring Recovery After Re-connect** from the **Reliability of Remote Monitoring** test suite – is used to give practical context of each part of the generalized process. Throughout the process, the example test case is presented to demonstrate how the plan document is constructed (Figure 4).

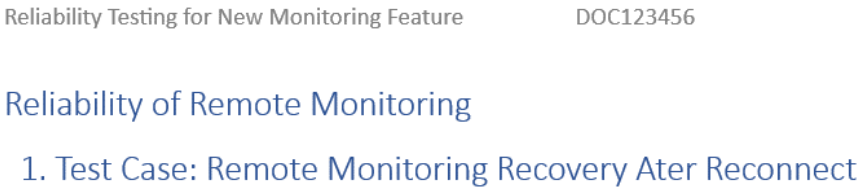

Figure 4. Test plan with unique document identifier and title.

It is worth noting that even if the example is grounded on real-world it is completely hypothetical. The process starts with the planning stage, where the test objectives need to be clear before continuing with the process.

4.2 Planning Stage

Well defined plan is crucial for ensuring testing achieves desired outcomes. The planning phase involves iterations of reviews with stakeholders to ensure test objectives are met. Planning is the process of defining test cases, each of which requires definition of following elements:

- Reliability metrics to monitor.
- Execution cycles or run time.
- Test setup and tools needed.

Each test case is added into the test plan document, with test results attached to it after test execution. The plan can be tailored to cover testing for a single feature or larger part of the system. When tests are used in regulation progress

– for example the validation of a new feature – it is preferred to exclude unrelated documents from that process. To account these cases, it is suggested to create a separate plan for the reliability testing of the feature instead of including it to any existing plans. This way documentation overhead and the need to revise old documents can be avoided.

4.2.1 Defining Test Objectives

Test planning begins by defining objectives for the testing and test cases based on them. Since the process has a top-down approach where the test objective is not defined by testers, but the focus is on the role of test engineer developing and executing the testing, risk management processes and test objective definitions are not explored in detail. The testing process is a combination of top-down and bottom-up approaches, leaving testers flexibility on the finer details such as tools and methodologies used.

Well-defined test objectives ensure correct focus enabling the rest of the process. To keep documentation concise, a single test plan can host multiple test cases. For example, testing done for a single feature should have one plan with all the relevant test cases included in it. In the example here, **Remote Monitoring Recovery After Re-connect** is just a one test case within test plan (Figure 5).

Figure 5. Test plan with multiple test cases.

In the top-down approach, test objectives are defined by using risk-based approaches by the designers, architects, product owners, developers, and other stakeholders who have the in-depth knowledge of the test target. They host meetings and workshops to gain insight into the fragile or error-prone parts ensuring comprehensive testing. The test objectives target identified weak points from risk management processes outlined in organizational risk management system, such as defined in ISO 14791 and further detailed in ISO/TR 24971 [14, 16]. Techniques such as Failure Modes and Effects Analysis (FMEA) are used to identify potential failure modes, their causes, and effects on patient and user safety.

First risk assessments are conducted to identify potential hazard associated within the defined intended use of the device. Once potential hazards have been identified, their severity and probability are assessed to help prioritization for evaluation. Risk evaluation compares these risks against acceptable level to decide which risks need to be addressed. For any residual risk in medical devices, mitigation needs to diminish its probability and severity as far as possible. Alternatively, the need for a new test case can emerge from continuous improvement actions. Reliability testing can act as the preventative action within Corrective and Preventative Action (CAPA) process mandated by regulation [3]. Testing needs to be expanded to cover possible weakness identified during the CAPA process.

Without a test objective, testing or even further planning is not possible. As it is the result of heavy process including several stakeholders and not necessarily the testers, it is usually the only non-editable part in this process. Other components, such as test setups and metrics, can be iteratively modified as needed later in the process. In the example test case – **Remote Monitoring Recovery After Re-connect –** the test objective is “Ensuring the recovery of clinical data flow after reconnect” (Figure 6).

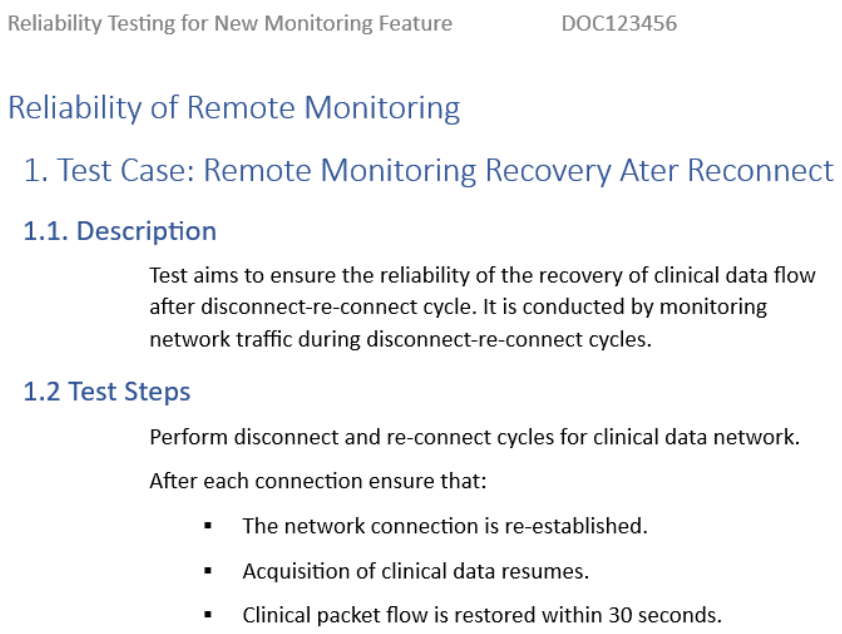

Figure 6. Test case with description of test goals.

With this example, the text continues to explore the reliability metrics in the next stage of the process.

4.2.2 Reliability Metrics and Run Time

After having received the identified test objectives, each test case needs to have clear metrics defined. In the process here the **specification** is the overall requirement that the test must meet, composed of metrics monitored during testing: **acceptance criteria, error rate**, and **run time**. These metrics are closely interrelated as illustrated in Figure 7.

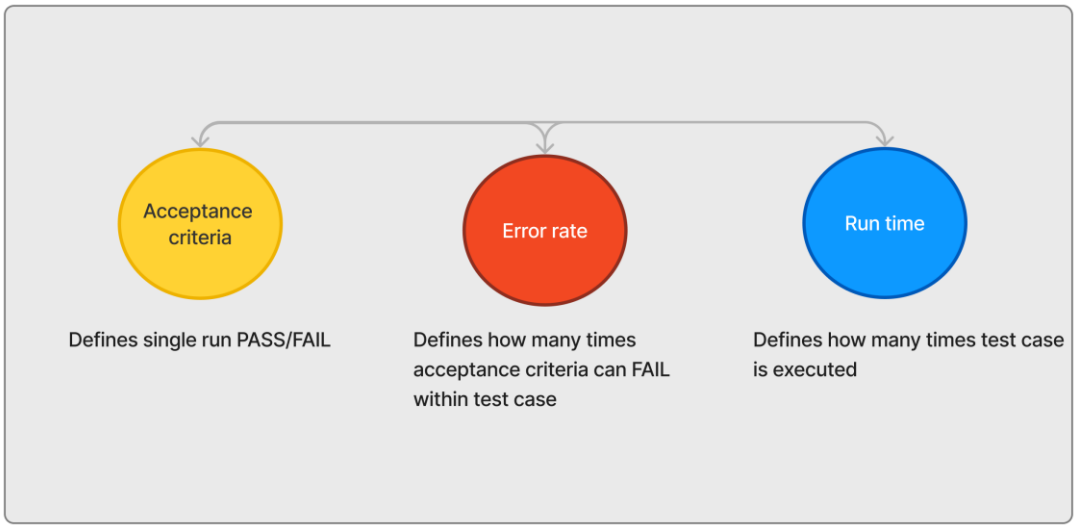

Figure 7. Metrics and their relation.

Defining accurate metrics is crucial, as the test development is based on them. This is especially vital with the long execution times of reliability tests. A closer examination of each metric is necessary to further understand their relationships and why they are so tightly bound together.

Acceptance Criteria

Acceptance criteria are the specific conditions that must be met within a defined timeframe during test execution. They set the benchmark against which the results are evaluated. In the context of patient monitoring system, acceptance criteria could be:

- Event response time: The sound of an event is heard within 800 milliseconds.
- System recovery time: Monitoring fully restored within 30 seconds of unexpected reboot.

The example test case – **Remote Monitoring Recovery After Re-connect** – has the acceptance criteria: “Network clinical data flow restored within 30 seconds after reconnecting”.

Error Rate

The **error rate** defines the accepted frequency at which a single execution of a test case can fail. This metric is critical in determining the required run time to gather sufficient data for statistical evidence of reliability. To ensure consistent reliability, pass/fail type acceptance criteria were supplemented with these statistical measures that address variability in results. These additional metrics can follow simple definitions such as **incidents per thousand exams (IPTE)** or established statistical quality control standards such as variations of the **Z-score** shown in Formula 1 [2, 24].

$$\mathrm{Z} = \frac{(\mathrm{X}-\mu)}{\sigma} \quad (1)$$

In this formula:

- X is value of single test result, the data point.
- μ is the mean of all results from test execution, the dataset.
- σ is the standard deviation of the dataset.

The Z-score represents the number of standard deviations a data point deviates from the mean in a normal distribution. For example, a Z-score greater than 4, tolerates only results that deviate from the mean by 0.003%, allowing little variation on the results between executions. In reliability testing, the Z-score can be calculated from the specified acceptance criteria – as the X value – to assess the deviation of the process from the specified limit to measure the consistency within the defined performance standard. For example, a connection test with 10 000 cycles can have defined maximum for acceptance criteria and Z-score for error rate. Alternatively, use of short-term Z-score, – **Zst** – where the dataset is smaller subset of the whole data, can assess the short-term variability. Short-term scores can be utilized to calculate long execution Z-scores from shorter test runs. Z-scores are a great tool at ensuring consistent behavior.

When error rate is defined as one incident per thousand exams (IPTE < 1), executing fewer than one thousand repetitions would be insufficient for reliable

conclusions. This kind of error rate directly impacts the number of test cycles needed to statistically validate that the system meets the acceptance criteria.

Combining the error rate, with the example test case and its acceptance criteria molds the specification to:

> "Network clinical data flow restored within 30 seconds after reconnecting with error rate of 0.5 incidents per thousand exams."

With the addition of the error rate, the test needs at least 2000 execution cycles without failure to reach it. In medical device testing, strict statistical thresholds such as "IPTE 0.5" or "Z-score ≥ 4" ensure high reliability, precision, and patient safety. Failure in reliability testing occurs if the test run fails to meet acceptance criteria more times than allowed by the defined error rate.

Run time

Run time is defined in terms of time duration or the number of test execution cycles. This decision considers the type of the test and defined error rate. The examples of test run times include:

- **Connection testing:** When introducing a new connection such as cable, run time can be 10,000 connection cycles with Z-score greater than 4.
- **Network reliability test:** Network clinical data flow restored within 30 seconds after reconnecting with error rate of 0.5 incidents per thousand exams. The error rate necessitates at least 2000 execution cycles.

The minimum run time is dictated by the error rate. In the network reliability test example, performing fewer than 2000 test cycles would be insufficient. Furthermore, run time impacts the acceptance criteria when using statistical measures like Z-score as their proof of consistency, increases together with every repetition of test.

Figure 8 shows the current state of the planning document after the reliability metrics have been defined.

Reliability of Remote Monitoring

1. Test Case: Remote Monitoring Recovery Ater Reconnect

1.1. Description

Test aims to ensure the reliability of the recovery of clinical data flow after disconnect-re-connect cycle. It is conducted by monitoring network traffic during disconnect-re-connect cycles.

1.2 Test Steps

Perform disconnect and re-connect cycles for clinical data network.

After each connection ensure that:

- The network connection is re-established.
- Acquisition of clinical data resumes.
- Clinical packet flow is restored within 30 seconds.

1.3 Test Metrics

Acceptance Criteria:

Network clinical data flow restored within 30 seconds after reconnecting

Error Rate:

IPTE < 0.5

Run Time:

2,000+ execution cycles

Figure 8. Test case with reliability metrics.

After the specification – with clear and objective metrics – defined, the foundation for the test development is set. Metrics guide the development, ensuring test focus on the reliability of the system, allowing the process to proceed to the test setup definition. Test setups are essential as they provide the context in which metrics can be measured and evaluated.

4.2.3 Defining Test Setups

Test setups define the testing environment ensuring the test objectives are met. The devices or features being tested are specified in the plan as **Units Under Test (UUT)**. Variations in hardware and software configurations can affect the

reliability metrics and test outcomes. When the test necessities specific tools to measure key reliability factors, they are defined in test setup.

Given the on metrics-driven reliability testing, it is essential to ensure that core components such as monitors, patient data modules, and simulators are included to provide real-time data acquisition needed for consistent performance assessment and measuring the reliability metrics. Patient data simulators provide input that mimic real-world patient conditions tying the test to real-world use.

In long-running reliability tests, test computers are utilized to execute the test scripts, allowing testing to be continuously monitored to prevent data loss during unforeseen failures. In robust test environments, these machines log all test data to a centralized database, ensuring data integrity throughout the testing. Additionally, external stress factors such as simulated network traffic overload can be included to mimic real-world stress factors, further enhancing the relevance of the test.

The test setups are organized according to devices under test, simulators, and external stress factors. Flexibility in the test setup allows this process to be adapted between different test cases, while maintaining core components for objective testing. In the example of **Remote Monitoring Recovery After Re-connect,** the test setup should include tools to monitor network traffic and provide clinical data. Network traffic capture would necessitate solution such as network switch with capture port and clinical data a patient data simulator (Figure 9).

Reliability of Remote Monitoring

1. Test Case: Remote Monitoring Recovery Ater Reconnect

1.1. Description

Test aims to ensure the reliability of the recovery of clinical data flow after disconnect-re-connect cycle. It is conducted by monitoring network traffic during disconnect-re-connect cycles.

1.2 Test Steps

Perform disconnect and re-connect cycles for clinical data network.

After each connection ensure that:

- The network connection is re-established.
- Acquisition of clinical data resumes.
- Clinical packet flow is restored within 30 seconds.

1.3 Test Metrics

Acceptance Criteria:

Network clinical data flow restored within 30 seconds after reconnecting

Error Rate:

IPTE < 0.5

Run Time:

2,000+ execution cycles

1.4 Test Environment

Test Setup:

Patient Monitor MODELXYZ

Tranport Setup

Network Capture Setup

Patient Data Simulator

Additional information:

Simulated Network Traffic

Figure 9. Test case with environment added.

To keep the plan short and concise, the setups for the whole testing can be defined in a separate document that is attached to the plan. This is useful if the test environments demand multiple devices, lengthening the test cases in plan. With the planning ready the next step is to move to the development stage.

4.3 Development Stage

The development phase covers all the necessary steps to enable the actual testing to take place. This includes gathering all the needed equipment, constructing the test setup, developing the automation code, and configure or create tools needed for the test. Test development is and iterative process

comparable to software development. Change needs and improvement opportunities can constantly be identified from dry runs and stakeholder feedback, leading to test revised as needed.

Collaboration with stakeholders is vital during the test development phase. The insights received help to ensure that developed tests align with the goals addressing the identified risks. The iterative nature and collaboration make this process development suitable for agile methodologies. The most notable and widely used example of modern-day agile work frameworks, especially in software development, is Scrum and the process suits scrum teams or testers working with stakeholders using it [25].

4.3.1 Methodologies and Tools

Selecting appropriate methodologies and tools is crucial to effective development and execution of testing. The focus during development should be on ensuring that the tests are accurately targeting the right things with desired outcomes.

Returning to the example test case – **Remote Monitoring Recovery After Re-connect –** where the objectives and specifications are now known, needed methodologies can be defined. The requirement is to be able to perform automated tests that monitor network data flow reliably. The test should be usable with different products and versions, if possible, and it should provide immediate feedback.

To achieve these requirements, several tools and technologies are needed:

- **Robot Framework** (RF) as the test automation framework.
- **Python** libraries manage more complicated logic and custom keywords needed for network traffic monitoring.
- **Docker** was used to provide isolated test environments allowing test execution in different environments and devices with consistent results.
- **Jenkins** for centralized automation server for test execution.
- **Grafana** to provide live monitoring.

- **GitLab** for version control.

Table 5 introduces each of these tools and briefly describes their purpose in testing.

Table 6. Tools and their descriptions.

Tool	Description
Robot Framework	Open-source, keyword-driven test automation framework. Provides a clear, modular way of writing reusable test scripts using high-level keywords.
Python	Has a rich ecosystem of libraries used for automation, testing, and development. Custom libraries manage complex data manipulation and extend RF capabilities.
Docker	Platform that enables developers to create, deploy, and run applications inside isolated containers, ensuring that code runs in the same environment across multiple systems. Highly valuable for maintaining consistent test environments.
Jenkins	Open-source automation server used to automate parts of software development like building, testing, and deployment. Integrated with CI/CD pipelines to trigger automated tests on code changes, providing centralized execution and results.
Grafana	Open-source platform for monitoring and observability. Visualizes data from multiple sources in customizable dashboards, crucial for real-time monitoring and analysis of long-running test metrics.
GitLab	Web-based version control tool that, allows iterative reviews and effective management of test code, scripts, and documentation.

Integrating these tools to test development phase leads to more robust and maintainable tests.

The reliability testing here uses Keyword-Driven Testing (KDT) by using RF. KDT improves modularity and test case maintainability by separating test logic

from tool-specific commands, making it particularly suited for medical device environments. Additionally, RF support the long test executions with repeatability, scalability and traceability. Combined with Python, it allows the creation of custom keywords that abstracts complex actions into high-level commands, for example “verify connection” or “check parameter value”. This modular test design allows components to be shared across test cases. Modularity adds enhanced reusability and maintainability, key factors in efficient test automation development. Python enables dynamic keyword libraries, able to adapt changes in system configurations. It offers flexibility, offering potential for custom keywords and libraries that extend the capabilities of RF while keeping the keywords and test code easily understandable.

Docker ensures that the test setups are consistent and reproducible, which is especially important in complex environments consisting of both hardware and software. By using these isolated and maintained environments, tests can be executed reliably across different systems.

Jenkins and Grafana provide tools for easier failure capturing when executing high number of test cycles. Additionally, they offer stakeholders immediate visibility to testing status and results.

By utilizing version control tools such as Gitlab, the communication during the testing process is enhanced enabling iterative early reviews throughout the development phase. Version control ensures proper management of test code, scripts, and documentation while supporting collaboration and traceability.

In the test case, – **Remote Monitoring Recovery After Re-connect** – RF is used to handle the disconnecting and re-connecting. Python is used for custom library which communicates with specifically setup network switch. This library would thread network traffic data from it to a parser on a different thread, capturing the necessary packets containing the clinical data and ensuring the acceptance criteria. The test needs Docker container to include test automation frameworks to communicate with the software of Unit Under Test (UUT). In addition, repositories with the test code and packet capture tools needs to be

included from GitLab into the Docker container. Jenkins is the platform where the test is executed on, providing live results and test reports. The Jenkins job would push metrics to Grafana after each execution cycle.

Writing reusable, modular, and maintainable test automation code is key for efficient test automation development. Automation helps reduce errors in repetitive tasks and enables long run times. Coupled with automated monitoring and error catching, this approach allows continuous execution outside working hours.

Reliability test developer needs to be aware of test optimization. With long execute times of reliability testing, even seemingly minor increase can significantly extend the overall testing time. For example, having 10 seconds of added execution time on each test cycle with 10 000 cycles, adds more than 27 hours to test run time. This is especially vital in larger systems with more tests, as the release needs to wait until planned testing has been completed.

4.3.2 Iterative Development with Dry Runs

Dry runs are an essential part of the test development process, allowing the validation and refinement of tests before official execution. A dry run is an unofficial execution of the test, conducted at any phase of the development, focusing on any part of the test. In test automation dry runs can range from simple syntax checks of test automation code to full or partial executions without official results.

In the context of the **Remote Monitoring Recovery After Re-connect** test case, several dry runs are conducted to ensure the automation code and custom libraries function correctly. These dry runs help to identify unforeseen interactions between different components, such as incorrect configurations or timing issues in the disconnect-reconnect cycles. For instance, a case where network traffic monitoring tool is not capturing data consistently can root from several different sources. There might be configuration error in the switch, error in the Python code, configuration error in Docker container, or error in RF code

not simulating data from the patient data simulator correctly. Additionally, as the process was for a new feature without existing data, the reliability metrics based have to be revisited based on observations from the dry runs.

By iteratively refining the test through dry runs and stakeholder feedback, the reliability and accuracy of testing process is increased. This iterative approach aligns well with agile methodologies, facilitating continuous improvement and adaptability.

The process does not define incident reporting as it relies on organizational practices for problem reports. Test case failures are managed through general defect processes, ensuring proper communication of essential information. By doing this the aim is to minimize disruptions and maintain focus while ensuring that no failure case persist as the release approaches.

4.3.3 Continuous Monitoring and Adjustments

Continuous monitoring of test executions and results is especially vital, especially in the development phase utilizing Continuous Integration and Continuous Delivery (CI/CD). While the focus is on reliability testing, performance data is simultaneously monitored, providing insights into the testing. Performance issues and system failures might only occur during these long test executions. For example, during an extended test run, the continuous monitoring can reveal a gradual increase in memory usage, indicating a potential memory leak in the software. In the testing, tools – such as Grafana – achieve real-time visualization of key metrics. By monitoring system response times and resource utilization, issues compromising the reliability of the system can be identified and addressed.

Continuous monitoring and test execution allows issues to be caught early, reducing risks and delays in the project. Providing stakeholders with access to continuous test execution data adds significant value for them. The use of data presentation tools allowed them to stay informed throughout the process.

Grafana for example, allows alerts and annotations to be added directly into the graphical visualizations, facilitating timely responses to issues.

By continuously monitoring, it is possible to detect errors, bottlenecks, and deviations within the testing. This iterative process with continuous monitoring, minimizes risks and increases testing, while performing well within moder day agile CI/CD environments.

4.4 Execution and Results

The execution phase is where planning and development efforts come into action. With thorough preparation, this phase involves running the tests, collecting data, and analysing the results to determine whether the system meets the reliability test objectives set earlier. To illustrate how this phase would proceed, the hypothetical execution of the **Remote Monitoring Recovery After Re-connect** test is included. Example demonstrates the practical application of the testing process, even though that the actual testing performed is not explained due to confidentiality reasons.

4.4.1 Test Execution

Once the setup was finalized and the test automation code was developed, the official test runs could stat for the example test case **Remote Monitoring Recovery After Re-connect**. The goal was to verify that the system could reliably recover remote monitoring functionality after a network disconnect-reconnect cycle, within the specified acceptance criteria. The tests were executed according to the defined run time and error rate specifications. The test automation frameworks of Robot Framework (RF) and Python executed the test steps, disconnecting and reconnecting the network connections while monitoring system responses and relevant metrics. During test execution the recovery time is monitored with each cycle, and it is compared to the acceptance criteria. If more failures than allowed by the error rate occur, testing is halted, and the issues analysed. Test runs need to be executed enough time

to gather sufficient data of the reliability, in this case at least 2000 error-free cycles to ensure specification of less than 0.5 incidents per thousand exams.

4.4.2 Data Collection and Documentation

Data collection was automated using the tools integrated during the development phase. The test automation scripts recorded the results of each test cycle, capturing detailed metrics. Grafana dashboard provided real-time visualization of the ongoing test, allowing to monitor progress and detect any issues. In case where the number of failures exceed the allowed error rate, the testing was configured to abort automatically to save time and resources, allowing immediate investigation.

Given the large volume of data generated over the execution cycles, manual documentation was impractical. The metrics are stored into centralized database, ensuring secure and organized data management and efficient analysis and reporting.

After the hypothetical test runs are completed, the collected data would be analysed by comparing the observed metrics against the predefined reliability metrics. The statistical analysis would confirm whether the system met reliability requirements. Relevant metrics, configurations and test result would be documented in the test plan. Detailed reports could be generated to support validation activities and regulatory compliance. Figure 10 shows the finalized test plan with the test execution results.

Reliability of Remote Monitoring

1. Test Case: Remote Monitoring Recovery Ater Reconnect

1.1. Description

Test aims to ensure the reliability of the recovery of clinical data flow after disconnect-re-connect cycle. It is conducted by monitoring network traffic during disconnect-re-connect cycles.

1.2 Test Steps

Perform disconnect and re-connect cycles for clinical data network.

After each connection ensure that:

- The network connection is re-established.
- Acquisition of clinical data resumes.
- Clinical packet flow is restored within 30 seconds.

1.3 Test Metrics

Acceptance Criteria:

Network clinical data flow restored within 30 seconds after reconnecting

Error Rate:

IPTE < 0.5

Run Time:

2,000+ execution cycles

1.4 Test Environment

Test Setup:

Patient Monitor MODELXYZ

Tranport Setup

Network Capture Setup

Patient Data Simulator

Additional information:

Simulated Network Traffic

1.5 Results

Test Comment: Executed as per plan

Results: PASS, Peetu Salonen, 28-Sep-2024

Figure 10. Final executed test plan document.

All metrics received, and needed configuration information, from the test execution are well documented either to results field in the test plan document, or to separate document that will be attached to it. Results can then be used to back up validation and prove the reliability of the system, device, or feature under testing.

5 Conclusion

The development and manufacturing of medical devices has the principle of patient safety as a top priority through the product's lifecycle. In the evolving landscape of medical device software, advanced tools and technologies have raised the standard for testing. Reliability testing, such as presented in this thesis, provides great insight on the overall condition of the software. By utilizing it issues and shortcoming can be revealed early, greatly increasing the confidence on new releases. The methodologies discussed are not limited to new features or even medical devices alone.

By implementing a structured approach to reliability testing, medical device manufacturers can ensure that testing is conducted in a manner that complies with industry standards and regulatory requirements. This allows reliability testing to be used for continuous improvement efforts and CAPA processes, playing critical role in the lifecycle management these devices.

As the medical device industry continues to evolve and new technology enters the field, the importance of robust testing practices cannot be overlooked. Reliability testing, integrated into the development process in a CI/CD environment offers a proactive way of identifying issues, reducing potential risks and ensuring the core value of patient safety.

Future improvements to the process could include more advanced data analytics, or machine learning by applying statistical methods to analyse test results over numerous executions and deployments. With this addition one could identify patterns or anomalies indicating of underlying issues. Additionally, exploring the use of Markov models – which have extensive literature available

on the subject – for reliability testing presents a promising are for future research.

The thesis provides practical insights into the implementation of and effective reliability testing in medical device system. Integrating robust testing practices with agile methodologies and modern tools, contributes to enhanced patient safety, advancing the field of medical device development.

References

1 ISO/IEC/IEEE 29119 (2013–2016). Software and systems engineering—Software testing. Geneva: International Organization for Standardization.

2 Ihalainen, Panu & Hölttä Taina. 2001. Six Sigma pähkinänkuoressa. Tampere: Tammer-Paino Oy.

3 Regulation (EU) 2017/745 of the European Parliament and of the Council. 2017. The official paper of European Union. European Parliament and Council.

4 Red Hat. What is CI/CD?. 2023. Online material. <www.redhat.com. <https://www.redhat.com/en/topics/devops/what-is-ci-cd>.

5 Robot Framework Foundation. 2024. Robot Framework Documentation. Online material. <https://robotframework.org >.

6 Python Software Foundation. 2024. Python Documentation. Online material. <https://docs.python.org/3/>.

7 Docker, Inc. 2024. Docker Documentation. Online material. <https://docs.docker.com/>.

8 Jenkins Project. 2024. Jenkins User Documentation. Online material. <https://www.jenkins.io/doc/>.

9 GitLab Inc. 2024. GitLab Documentation. Online material. <https://docs.gitlab.com/>.

10 Grafana Labs. 2024. Grafana Documentation. Online material. <https://grafana.com/docs/>.

11 Duodecim. 2023. Akuuttihoitotyön opas (aik. Teho- ja valvontahoitotyön opas). Duodecim Terveysportti.

12 Huusko, Juhamatti; Kinnunen, Ulla-Mari & Saranto, Kaija. 2023. Medical device regulation (MDR) in health technology enterprises - perspectives of managers and regulatory professionals. BMC health services research, 23(1), 310.

13 Business Finland. 2020. European Medical Device Regulations MDR & IVDR – A Guide to Market. Online material. <https://www.leanentries.com/wp-content/uploads/european-medical-device-regulations-mdr-ivdr-a-guide-to-market.pdf> Read 23.3.2024

14 Suomen Standardisoimisliitto SF ry. 2019. Medical devices. Application of risk management to medical devices (ISO 14971:2019). SFS Online.

15 Suomen Standardisoimisliitto SF ry. 2016. Medical devices. Quality management systems. Requirements for regulatory purposes (ISO 13485:2016). SFS Online.

16 Suomen Standardisoimisliitto SF ry. 2020. Medical devices. Guidance on the application of ISO 14971 (ISO/TR 24971:2020). SFS Online.

17 ISO a practical guide, ISO 13485:2016 Medical devices, Advice from ISO/TC 210. 2017.

18 Rittgers, J. 2021. An Approach to Reliability Growth for Medical Device Development. Annual Reliability and Maintainability Symposium (RAMS) 2021, pp. 1-6. Orlando, FL, USA.

19 Bhanushali, Amit. 2023. Ensuring Software Quality Through Effective Quality Assurance Testing: Best Practices and Case Studies. International Journal of Advances in Scientific Research and Engineering 26.1.

20 Abd Rahman, N. H.; Ibrahim, A. K.; Hasikin, K & Abd Razak, N, A. (2023) Critical Device Reliability Assessment in Healthcare Services. Journal of healthcare engineering. Online material. <https://doi.org(10.1155/2023/3136511/>. Read 24.9.2024

21 Islam, Girbrail & Storer, Tim. 202. A Case Study of Agile Software Development for Safety-Critical Systems Projects. Reliability Engineering & System Safey. Vol. 200.

22 Zampetti, Fiorella; Tamburri, Damian; Panichella, Sebastiono; Panichella, Annibale; Canfora, Gerardo & Di Penta, Massimiliano. 2023. Continuous Integration and Delivery Practices for Cyber-Physical Systems: An Interview-Based Study. ACSM Transactions on Software Engineering and Methodology. Vol 32, Issue 3.

23 Magrupova, M, T; Matyakubove, P, M & Abdihalikov, S, P. 2020. Methods for Increasing and Assessing Reliability of Medical Equipment. International Conference on Information Science and Communications Technologies (ICISCT), 2020, pp. 1-3. Tashkent, Uzbekistan.

24 Christmann, P, Edwinn. 2012. Beyond the Numbers Making Sense of Statistics. NSTApress.

25 Scrum.org. 2023. Scrum. Online material. <https://www.scrum.org/>. Read 20.4.2024.