FeatureCloud

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Privacy preserving federated machine learning and blockchaining for reduced cyber risks in a world of distributed healthcare

Deliverable D3.8
"Manuscript on Usability process"

Work Package WP3
"Guidelines, standardization, and certification"

Disclaimer

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078. Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Copyright message

Document information

Grant Agreement Number: 826078				**Acronym: FeatureCloud**
Full title	Privacy preserving federated machine learning and blockchaining for reduced cyber risks in a world of distributed healthcare			
Topic	Toolkit for assessing and reducing cyber risks in hospitals and care centres to protect privacy/data/infrastructures			
Funding scheme	RIA - Research and Innovation action			
Start Date	1 January 2019	**Duration**	60 months	
Project URL	https://featurecloud.eu/			
EU Project Officer	Christos Maramis, Health and Digital Executive Agency (HaDEA)			
Project Coordinator	Jan Baumbach, University of Hamburg (UHAM)			
Deliverable	D3.8 - Manuscript on Usability process			
Work Package	WP3 - Guidelines, standardization, and certification			
Date of Delivery	**Contractual**	31/07/2023 (M55)	**Actual**	31/07/2023 (M55)
Nature	Report	**Dissemination Level**	Public	
Lead Beneficiary	02 UMR			
Responsible Author(s)	Dr. Dominik Heider (UMR) Dr. Anne-Christin Hauschild (UMR) Dr. Roman Martin (UMR) Sandra Clemens (UMR) Vanessa Klemt (UMR) Dr. Joachim Wienbeck (External Advisor)			
Keywords	Usability guideline, MDx, knowledge transfer from research to industry			

Table of Contents

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

1 Table of acronyms and definitions

CDSS	Clinical Decision Support System
concentris	concentris research management gmbh
FDA	Food and Drug Administration (United States of America)
FTA	Fault Tree Analysis
D	Deliverable
GDPR	General Data Protection Regulation
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
MDCG	Medical Device Coordination Group
MDR	Medical Device Regulation
MDx	Molecular Diagnostics
MS	Milestone
Patients	In this deliverable, we use the term “patients” for all research subjects. In FeatureCloud, we will focus on patients, as this is already the most vulnerable case scenario and this is where most primary data is available to us. Admittedly, some research subjects participate in clinical trials but not as patients but as healthy individuals, usually on a voluntary basis and are therefore not dependent on the physicians who care for them. Thus, to increase readability, we simply refer to them as “patients”.
PMS	Practice Management System
RM	Risk Management
SaMD	Software as a Medical Device
SRS	Software Requirements Specification
SLC	Software Life Cycle
SOUP	Software of Unknown Provenance
UE	Usability Engineering
UI	User Interface
UOUP	User Interface of Unknown Provenance
UHAM	University of Hamburg
UMR	Philipps Universität Marburg
WP	Work Package

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

2 Objectives of the deliverable based on the Description of Action (DoA)

The objective of Work Package 3 (WP3) is to develop guidelines for a standardized academia-tailored software development process and to compile a documentation guideline for molecular diagnostics (MDx)-ready software (**Objective 1**), allowing a successful transition from academia-driven projects into MDx feasible. The overlying aim of these guidelines and recommendations is to maintain high-quality software and reduce the crucial error rate for medical-related software. As a follow-up to the other guidelines that addressed the Quality Management (D3.3), Software Life Cycle (D3.4), and Risk Management process (D3.6), this particular manuscript is dedicated to **Task 4**, the Usability process. Here, we present a streamlined Usability Process for academia-tailored development derived from the International Electrotechnical Commission (IEC) 62366. This Deliverable also completes milestone 22, titled “Concrete suggestions to fulfil MDx requirements for usability management” (**MS22**). The manuscript will be complemented with the concrete implementation of such a process for the FeatureCloud project in the Fifth Explicit Quality Control (D3.9). Finally, all guidelines will be made publicly available to provide the same standards for software that will be developed on top of FeatureCloud by third parties.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

3 Executive Summary

3.1 Methodology

Regulatory requirements for medical devices in terms of usability are extensively described in the IEC 62366. The Medical Device Regulation (MDR) highlights the importance of an adequate usability process since usability strongly affects risk elimination and ensures proper functionality for the user. Complementary to that, the Food and Drug Administration (FDA) of the United States of America emphasizes usability as the factor that incorporates human activity and usability engineering to optimize the medical device design. Beyond the regulations, there is so far little guidance on how to effectively develop software as a medical device that meets these requirements. Therefore, we analyzed the critical factors of the requirements and proposed a guideline that allows an academia-tailored implementation of the medical norm for research institutes. Our guideline delivers guidance for a smooth technology transfer from research to industrial standards.

3.2 Main results

With this deliverable, we proposed introducing a more feasible usability process and usability engineering process for academia and research institutes based closely on the current medical device standards, allowing the realization of MDx-ready implementation with manageable expenses. We condensed and streamlined the most important critical points of the required standards to a minimal amount of essential process activities into our suggested guideline, which should be fulfilled. Further, the guideline focuses on the usability concept, implementation, risk management, and optimization through various process activities such as use specification, use-related risk analysis, formative and summative evaluation, and user interface specification. By following our guidelines, a feasible usability process can be realized by research institutes that will result in the creation of documents that allow possible technology transfer to industrial organizations since our recommendations are based on industrial standards.

3.3 Progress beyond the state-of-the-art

Achieving a medical device certification, including Software as a Medical Device (SaMD), requires fulfilling all medical device regulations (MDR) through multiple different International Electrotechnical Commission / International Organization for Standardization (IEC / ISO) standards, such as ISO 13485 for quality systems, ISO 14971 for risk management, IEC 62304 for software life cycle or IEC 62366 for usability. In fact, for small research groups that often exist in academia, it is not feasible to fulfill all these requirements. Instead, most groups tend to create a proof-of-concept prototype that often contains obstacles to a potential transfer to industrial partners. This problem is even more dramatic since research projects are often carried out mainly by someone employed over short fixed-term contracts (Riemenschneider et al. 2018). To remedy this issue, we created a guideline for academia and research institutes development that requires less effort and, thereby, is more feasible to realize. Together with our guidelines for quality management, software life cycle, and risk management, we are lowering the barrier for technology and knowledge transfer to industrial partners.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

4 Usability Engineering Guideline

4.1 Definition of Usability and Usability Engineering

Usability is a key element in all types of medical devices because every product, from physical devices to independent software systems, has a user interface that allows human-machine interaction (Hastenteufel and Renaud 2019). The term usability should not be confused with user experience. Usability is related to the difficulty of using a medical device. On the contrary, the concept of user experience comprises the users' expectations prior to use, the subjective perception and feelings when interacting with the medical device as well as the satisfaction after the usage. In other words, usability is only one component of the overall user experience (Geis and Johner 2015).

ISO 9241 defines usability as the extent to which an interactive system can be used by certain users in the intended context of use in order to achieve specific goals effectively, efficiently and with user satisfaction. This involves every step from installation to execution. Thereby, effectively means that users are able to achieve specified goals completely, correctly and accurately. For example, software that fulfills all stakeholder requirements can be used effectively. On the other hand, efficiently means that the goal is achieved with reasonable and minimal resource consumption. (Geis and Johner 2015).

The definition of the IEC 62366-1 differs a bit. According to IEC 62366-1, "Usability is created by characteristics of the user interface that facilitate use, i.e., to make it easier for the user to perceive information presented by the user interface, to understand and to make decisions based on that information, and to interact with the medical device to achieve specified goals in the intended use environments" (IEC 62366-1 2015). According to the norm, usability measures include effectiveness, efficiency, user satisfaction, learnability, and memorability. All of them can affect the safety of the medical device. The term usability also includes aspects such as the aesthetics of the user interface, which are not directly related to the safety of the device. However, the IEC-62366-1 does not focus on overall usability but rather on the above-mentioned safety-related aspects of usability in order to reduce use-related risks (Hastenteufel and Renaud 2019).

Both definitions still correspond to the notorious usability model of Nielsen from 1994 which is demonstrated in Figure 1. Nielsen defined four usability characteristics and added concrete metrics to measure each aspect (Nielsen 1994).

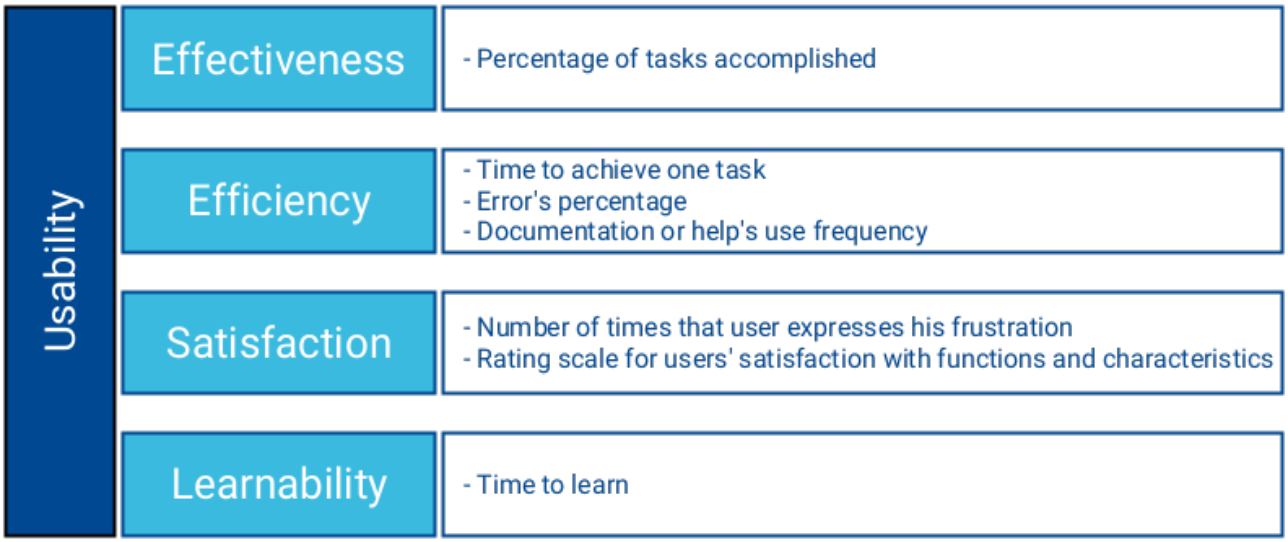

Figure 1. *Usability Model.*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

To conclude, all definitions agree that the usability of software is not an absolute quality property but is highly dependent on the characteristics of the users, the intended goals, and the use environment. For this reason, usability can only be assessed during the actual use of the software (Mentler 2018). Thus, developing interactive systems with good usability requires a continuous and consistent focus on the intended users, their needs, goals, and tasks and then involving them when iteratively evaluating the user interface (Hastenteufel and Renaud 2019).

To do so, usability engineering is an applied discipline that uses systematic and user-oriented methods to achieve usability in the design of user interfaces (Mentler 2018). Usability engineering can also be referred to as Human factors engineering. The IEC 62366-1 defines usability engineering as the "application of knowledge about human behavior, abilities, limitations, and other characteristics to the design of medical devices (including software), systems, and tasks to achieve adequate usability". The fundamental thinking and procedure associated with usability engineering and user experience design are also summarized by Butler in a cycle of analysis, design, implementation, and evaluation, shown in Figure 2. Thereby, with each iteration, the usability of the user interface increases (Butler 1996).

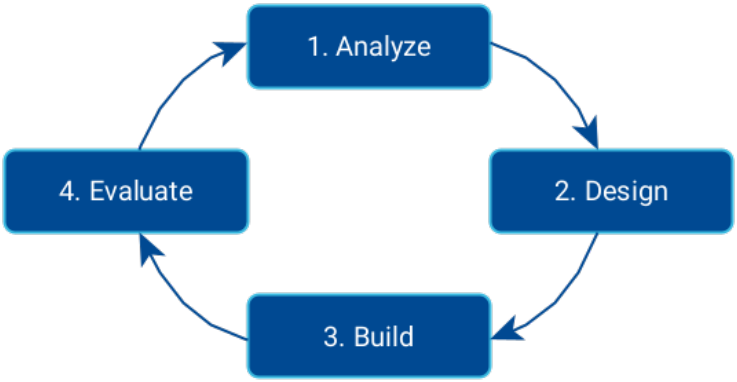

Figure 2. *Usability Engineering Paradigm.*

This paradigm is also reflected in the international IEC 62366-1 norm, which describes a usability engineering process for designing and developing user interfaces. The process also puts the human at the center of considerations and implies an iterative UI design procedure. Therefore, following the process ensures that the resulting user interface prevents potentially harmful use errors so that use-associated risks of the software are reduced to acceptable levels. To sum up, to achieve good usability, implementing a process like this is inevitable.

4.2 Process Overview, Integration and Differentiation

The IEC 62366-1 demands that medical device manufacturers follow a UE (Usability Engineering) process during the user interface development of medical devices, including SaMDs (IEC 62366-1 2015; Mangul et al. 2019). The process comprises all steps required to "analyze, specify, design and evaluate the usability of a medical device as it relates to safety". It aims to systematically reduce risks related to human use errors to acceptable levels and enhance usability (IEC 62366-2 2015). In doing so, use errors shall be identified and eliminated by implementing suitable risk control measures on the user interface to deal with users' unpredictability (Lipprandt and Röhrig 2018).

The UE process can help to identify but does not mitigate risks associated with intentional misuse of the SaMD also referred to as abnormal use. Its focus rather lies on the normal use of the medical device, which includes correct use as well as use errors. Use errors can accidentally occur as the user misunderstands or misreads user interface elements. The process also supports the production of an intuitive and user-friendly interface depending on the feedback received throughout iterative evaluations (Shaheen et al. 2021).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

However, a full-blown usability engineering process is not feasible for academic software projects constrained by limited resources. For this reason, the regulatory requirements on usability engineering have been confined. As a result, the following guideline describes an academia-tailored usability engineering process that will lower the hurdle for research organizations to develop as close to the international standard IEC 62366-1 as reasonably possible. Consequently, the guideline has the potential to greatly facilitate and speed up the technology transfer to industrial manufacturing. Moreover, to build a bridge between the abstract norm and specific implementations, the regulatory requirements on usability have been complemented by further information, explanations, and concrete recommendations from the literature review.

The proposed guideline is centered on procedures that can be structured into four key phases: use specification, use-related risk analysis, iterative design cycle, and summative evaluation. Within each phase, specific activities shall be carried out (Janny and Pfeffer 2020). In addition, some suitable methods are suggested within the guideline. However, the selected methods and tools to perform the process may deviate in a specific project, depending on the planned user interface's complexity and the potential harm's severity. Moreover, the most critical role that academic organizations should assign for IEC 62366-1 is a usability engineer. Usability engineers do not have to be designers; they rather have the expertise and methodological knowledge to design and evaluate user interfaces (Geis and Johner 2015). They should ensure that the required usability engineering procedures described within this guideline are considered during the software development.

Furthermore, a usability engineering file that collects all the results produced by the process is the central element in providing evidence that compliance with the norm has been achieved. This document should at least contain references to all required documents. Research organizations usually need to pay more attention to formal documentation and procedures. Nevertheless, reliable, well-structured, and complete documentation facilitates the technology transfer from academia to the industry. For this reason, the guideline provides recommendations on how to manage documentation while keeping the effort manageable. Figure 3 provides an overview of the usability engineering process with its key activities and outputs that will be further concretized in this guideline. The UE process demonstrates how the different activities intertwine and build upon each other. Therefore, it is essential to understand that usability engineering activities can follow a more relaxed timely order. Instead, they can be performed in a sequence appropriate to the specific project circumstances (Johner, Hölzer-Klüpfel, and Wittorf 2021).

By definition, the UE process is a risk management process for the user interface. The main difference is that the risk management process, as defined in ISO 14971, takes the overall view of potential software and component failures to determine the acceptable risk of the SaMD, whereas the UE process is focused on reducing risks of the user interface associated with human use errors. Moreover, unlike the UE process, the risk management process addresses abnormal use, such as reasonably foreseeable misuse. These two processes intertwine, and corresponding efforts should be integrated into practice. The four key information flows between the processes are explained in the following (a to d) and referenced in Figure 4**.**

A. The use specification from the usability engineering process can be an input for the intended use statement, a prerequisite of risk management according to ISO 14971. In general, the required content of both documents is mostly similar and should be congruent.

B. The results of the use-related risk analysis are an input for the risk management process's risk analysis. It is recommended to store the results of both analyses in a shared risk sheet. This information flow is essential because some use-related risks might need further risk control outside the user interface.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

C. The identified hazards and sequences of events leading to hazardous situations from ISO 14971 are additional inputs for the UE process, particularly to determine hazard-related use scenarios.

D. After a successful summative evaluation, it is the responsibility of the risk management process to evaluate whether the overall residual risk is at acceptable levels, including risks posed by software failures and those from use errors (ISO 2019). Therefore, the UE process would provide (if available) the collected statistical data from the summative evaluation, which can be used to estimate the probability of use-related risks. Moreover, usability engineers should justify why further risk reduction through user interface improvement is not practicable.

After that, the UE process is completed, whereas the risk management process continues to monitor and review risks during further software development and ends with the potential final knowledge transfer to the industry.

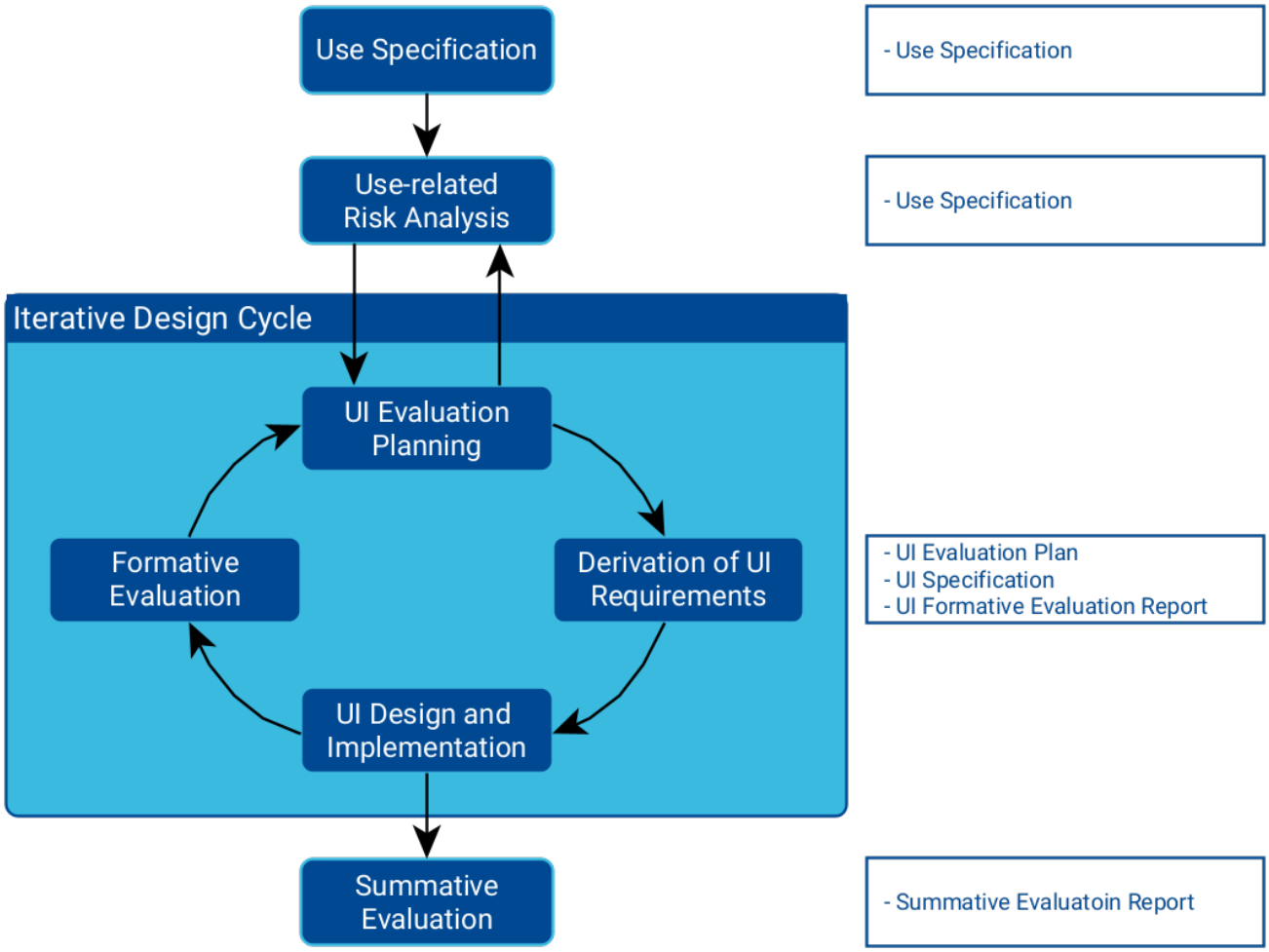

Figure 3. *Usability Engineering Process with resulting documents (right-hand side).*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

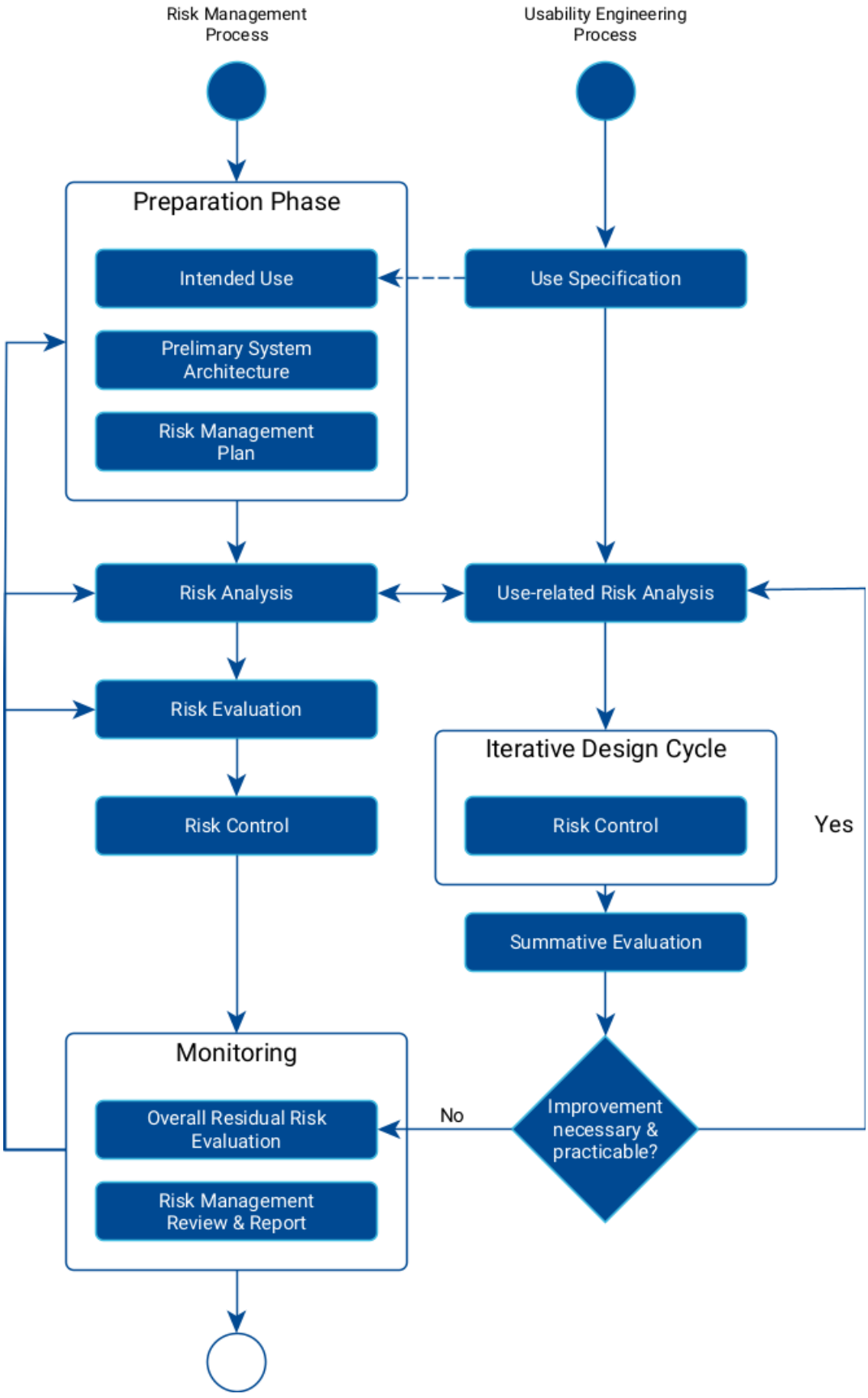

Figure 4. *Combination of academia-tailored risk management and usability engineering processes with the relevant interfaces between both.*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

4.3 Use Specification

A statement of intended use for the prospective software must be provided at the beginning of a new project. The term “intended use” or synonymously “intended purpose” is defined by the MDR as the use for which the SaMD is intended according to the manufacturer (MDCG 2019). Defining the intended use is an essential basis for further activities in the development and approval of SaMD. It determines whether the software might be qualified as a medical device in the future and, thus, whether the regulations apply. In addition, the risk classification of the SaMD according to the MDR is also based on the intended use statement (Beckers, Kwade, and Zanca 2021).

Moreover, the risk management process presupposes this statement of intended use. The statement can also receive further input from the use specification established at the beginning of the usability engineering process. In general, a clear differentiation between those two documents is difficult to make, and often the use specification is integrated into the intended use statement (Johner, Hölzer-Klüpfel, and Wittorf 2021).

The use specification can be seen as the cornerstone of specifying and designing a usable user interface of software (ISO 9241-11 2018). Therefore, in academic projects, the essential characteristics within the context of the use of the software shall be discussed in the beginning and documented in the use specification. The first version can be very high-level, including a preliminary summary of user groups, use environment, and medical indications. The document must be reviewed and updated during the UE process as more knowledge is gained, e.g., through additional user research or formative evaluations. According to the IEC 62366-1, the use specification shall contain at least information on the following aspects:

1. Intended medical indication
2. Patient population
3. Parts of the human body or type of tissue applied to (less relevant for SaMD)
4. User profile
5. Use environment
6. Operating principle (less relevant for SaMD)

The technical report IEC 62366-2 recommends several methods for developing the use specification. For example, context analysis is a common technique to gain information on the intended users, the user environment and the user’s tasks. The context analysis is either an observation or a one-on-one interview with a representative user, both conducted within the users’ workplace. For a structured meeting, the moderator should prepare guiding questions in advance (Geis and Johner 2015). Alternatively, in-depth group or one-on-one interviews with relevant stakeholders of the project, as well as survey techniques, might be less time-consuming. They can easily be conducted online, target a larger audience, and help gain insights into the user’s needs, perceptions and opinions (IEC 62366-2 2015). Stakeholders are all individuals or organizations that are interested, involved or affected by the software development project, e.g., intended users, patients, developers, and project managers (Geis and Johner 2015). Existing data on the intended users could also be used if available (Lipprandt and Röhrig 2018). Depending on the available time and resources of the academic project, the most suitable method to gain information on the intended use may differ.

The intended medical indication of the SaMD can, for instance, be screening, monitoring, treatment, prediction, or diagnosis of specific diseases, injuries and/or disabilities. The medical purpose can be defined either broadly or narrowly, depending on whether academic researchers can already foresee an exact medical use case. Characterizing the patient population by age group, gender, weight range, health condition, and disabilities has to be done in the case of physical medical devices. However, it is less important when dealing with software because standalone software is not directly interacting with the human body. Nevertheless, whether said information is crucial for developing the user interface should depend on the specific software.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

Furthermore, by applying research methods, information on the prospective users can be gained. The intended users can be subdivided into user groups as there might be more than one user group of the prospective software. After identifying the user groups, it is necessary to describe all their characteristics likely to influence usability and thus affect user interface design decisions. Creating user profiles is essential because an optimal UI meets the users' capabilities and needs. Users' characteristics considerably impact the efficiency, effectiveness, and safety of the SaMD. For example, a specialized physician has different knowledge than unskilled patients as to why they might be able to operate the SaMD differently. By considering user research results, the user interface can be designed to enable the intended users to use the device without making errors that compromise medical care or patient safety (FDA 2016). User Profiles can be documented in tables, in text form, or by creating personas (Geis and Johner 2015). To describe the user profile, one can choose a subset of the following aspects:

- User group (e.g., clinicians, patients, IT personnel)
- Occupation, expertise and education
- Demographic traits (e.g., gender, age group, language)
- Disabilities (e.g., limitations due to vision, hearing, cognitive impairments)
- User tasks within the domain or business process that the software shall support

As mentioned in the last bullet point, at early stages, it is helpful (but not mandatory) for the academic developers to define anticipated tasks which the users should be able to perform on the prospective user interface. Collecting those tasks can help create a general sense of how users will interact with the planned software. This elicitation of core tasks marks the starting point for defining use scenarios during the risk analysis. Core tasks are typically described using nouns and verbs, e.g., "check the number of migraine days and adjust medication".

The use environment describes the physical and social environment in which the software shall be used. This includes all conditions and settings in which users interact with the SaMD. Especially the conditions that determine optimal user interface design should be identified. The use environment can be described specifically for each user group as part of the user profile. There might be more than one possible use environment for software. For each use environment, the following aspects could be concretized:

A. Physical: clinical setting (e.g., hospital, laboratory) or non-clinical setting (e.g., home, outdoors, office), noise levels, temperature, lighting

B. Social: team work, stress level, frequency of use, distractions

Furthermore, in the case of hardware medical devices, the norm requires specifying the operating principle, which would be the underlying technology and mechanisms by which the medical device works. For stand-alone software, this is less relevant to define. Nevertheless, one can describe the inputs into the software (e.g., processing data) and the software outputs (e.g., returning diagnosis).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

4.4 Use-related Risk Analysis

The use-related risk analysis is a key component of usability engineering activities. The IEC 62366-1 requires several steps which define a complementary procedure to the risk management process according to ISO 14971. For this reason, it is recommended to document the results of the use-related risk analysis and those of the risk management process in a shared and comprehensive risk sheet. Regarding the realization of the risk analysis, the IEC 62366-1 only determines the required results. Beyond that, it is up to the academic project team to decide how to achieve those results.

A first draft of the use specification must exist to conduct a use-related risk analysis because possible risks are highly dependent on the medical context of use. Furthermore, due to the innovative character of academic software projects and often limited knowledge of the clinical utility, it can be challenging to foresee potential risks of the device design and the interaction with the user in the early stages (Beckers, Kwade, and Zanca 2021). On the other hand, if a similar or predecessor software product is already on the market, reported problems should be analyzed based on customer complaints or authority databases, collecting use errors. However, a preliminary user interface risk analysis can usually be challenging and time-consuming for academic projects. Instead, usability problems, including use errors, hazards, hazardous situations, and hazard-related use scenarios, could be more easily detected during the development and continuous formative evaluations as the user interface design matures and the intended medical purpose is further specified. Thus, risk assessment and control are ongoing activities throughout the UE process. Updating the risk sheet as new usability problems are identified is essential. The following steps will be explained to obtain the information needed for the use-related risk analysis.

At first, the IEC 62366-1 requires the identification of safety-related UI characteristics. By definition, a safety-related UI characteristic is an operating function that is required at the user interface and can affect the safety of the SaMD. Such an operating function can be a control element or any information on the UI with which the user interacts to perform a task on the software. To determine the safety-related features of the user interface, it is necessary to determine those operating functions that support critical tasks that can lead to severe damage if performed incorrectly or not at all. Design shortcomings of these operating functions are problematic because they may cause use errors, which can trigger a hazardous situation leading to significant harm (Wood 1998).

As already indicated, with the focus on safety-related features, potential use errors that might occur when the user interacts with the software must be identified. A use error is an incorrect user action or lack of necessary action that leads to an undesired result (IEC 62366-1 2015). For example, when the user makes an incorrect entry or selects the wrong button (Geis and Johner 2015). Use errors pose an unacceptable risk to the patient if they can potentially cause a hazardous situation that can harm a person's health. Use errors often result from unfavorable user interface design. Therefore, the primary goal in usability engineering is to prevent possible use errors by implementing risk control measures and thus achieving an intuitive and easy-to-learn user interface that meets the needs of the intended user group. Several usability engineering techniques can be applied in formative evaluations to discover possible use errors. At the early stages of the project, a preliminary list of potential use errors could be identified through discussions in focus groups, inspections, or expert reviews. This list will most likely also contain use errors that are not safety-related (IEC 62366-2 2015). As the software matures, the most accurate method is to observe use errors as representative users interact with the software during usability tests (Ravizza et al. 2023). In general, to identify foreseeable use errors, factors in the context of use (see use specification) should be considered, such as users' workload or health condition (Johner, Hölzer-Klüpfel, and Wittorf 2021).

Table 1 suggests a structure that can be used to document identified safety-related UI characteristics, corresponding potential use errors and an explanation of the root cause. Understanding by what a use error was caused (e.g., user oversees, misreads or misinterprets

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

something), allows to reveal design shortcomings and identify required modifications to improve the UI design.

Table 1. *List of safety-related UI Characteristics and potential Use Errors.*

ID	Safety-related UI Characteristics	Use Error	Root Cause

In the next step, the IEC 62366-1 demands to analyze all possible consequences that can be triggered by human use errors (IEC 62366-2 2015). Therefore, based on the identified use errors, a list of foreseeable hazards and hazardous situations, that might result from use errors, has to be created (see Table 2) (Lipprandt and Röhrig 2018). A hazard is a potential source of harm (e.g., wrong medication, treatment or prediction) and a hazardous situation is a condition where people are exposed to this hazard (e.g., patient receives wrong or too high medication) (IEC 62366-1 2015). Depending on the sequence of events, users can be exposed to a specific hazard in different hazardous situations and each hazardous situation can result in different types of harm (ISO 14971 2019). Besides using evaluation methods, a risk management workshop is a common way to identify hazards. A team approach has the advantage that several experts with different viewpoints can brainstorm and discuss potential use errors and the harm that could result from them (FDA 2016). The same analysis is also done within the risk management process described in ISO 14971. The exchange of the results from both processes is necessary and given by documenting all risks in one risk sheet, as mentioned before.

Table 2. *List of Hazards & Hazardous Situations.*

ID	Hazard	Hazardous Situation

Based on the identified use errors and hazardous situations, hazard-related use scenarios must be identified and documented. Hazard-related use scenarios describe sequences of user actions on the SaMD in which at least one use error occurs that potentially results in a hazardous situation that compromises medical care and causes damage to the health of patients (harm). Hazard-related use scenarios can help foster a common understanding of what might go wrong during the use of the software and are, therefore, key to systematically mitigating user interface risks. There are different methods to identify these use scenarios: literature review, one-on-one interview, brainstorming use scenarios, conducting a Fault Tree Analysis (FTA), or a Failure Mode and Effect Analysis (FMEA). The FTA and FMEA are among the most widely used risk analysis techniques. They both presuppose that the software architecture has been defined. More information on how to apply these methods can be found on the web pages of the Johner Institute.

The documentation of hazard-related use scenarios shall at least include a sequence of user tasks supported by the SaMD, the resulting harm, and its severity level. To achieve that, this guideline proposes to break down each critical core task into a sequence of sub-tasks that describe the concrete workflow of user actions on the UI (Barredo Arrieta et al. 2020). This can be pretty challenging at the early stages of developing novel software for which no reference examples exist. For this reason, it is easier to first describe use scenarios on a high-level and then add tasks and details as the user interface design matures throughout the design process (Elahi 2018). The following table structure, as shown in Table 3, is one solution to document hazard-related use scenarios properly. In case of more than one user group, the described use scenarios must be assigned to the respective user group that intends to perform the associated tasks. All in all, the use

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

scenarios link all the individual components of the risk analysis and then associate a risk control measure that resolves the usability problem. The UI specification in Chapter 3.4.2 will explore identifying these risk controls.

Table 3. *Hazard-related Use Scenario Description*

ID: Core Task	• User group • Preliminary condition (initial situation before start the task) • Post condition (target work result)			
Sub Task	Use Error (ID)	Hazardous Situation (ID)	Harm & Severity Level	Risk Control Measure

As mentioned, for each possible use error that could lead to harm, the IEC 62366-1 requires documenting the severity of the associated harm within the hazard-related use scenario. Severity is a “measure of the possible consequences of a hazard”. In order to treat risks consistently, the severity levels (SL) should follow the categorization scheme defined within the risk management process of ISO 14971. A common scheme is presented in Table 4. Each level is mapped to the possible consequences of the harm (Johner, Hölzer-Klüpfel, and Wittorf 2021). Severity estimates can be obtained from the prior analysis of hazardous situations.

Table 4. *Severity Levels of Harm.*

Severity Level of Harm	Description
Negligible	Results in inconvenience or temporary discomfort, no injury
Minor	Results in temporary injury not requiring professional medical care
Serious	Results in injury or impairment requiring professional medical care
Critical	Results in permanent impairment or life-threatening injury
Catastrophic	Results in patient death

The final list of hazard-related use scenarios - which will not be achieved until after the completion of all formative evaluations - shall be used in summative evaluation to ensure that the evaluation focuses on the tasks related to safe use. It can be decided whether summative evaluation includes all hazard-related use scenarios or only a subset of them. It is essential to document which hazard-related use scenarios have been selected and, if applicable, based on which selection scheme. The IEC 62366-1, for example, proposes to select hazard-related use scenarios based on “the severity of the potential harm that could be caused by a use error”. Depending on the risk policy defined in ISO 14971, a subset could include all potential use errors that lead to harm or only those use errors that cause a severity level of 3 to 5. The risk policy determines the extent to which the risk of the specific software can and should be reduced.

4.5 Iterative Design Cycle

Many research groups follow an iterative design and development process for the user interface. Establishing an iterative process like that is also recommended by this guideline, as it ensures that usability concerns will be considered and assessed as new features are incorporated into the UI design. The UI development process is independent of the structure of the chosen software

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

development model, which can be linear or iterative (e.g., Waterfall, V-Model, Scrum). The UI design cycle starts with the first version of the user interface specification, which contains the requirements for the user interface and thus provides fundamental design inputs. In practice, specifications and design are often developed hand in hand. Especially in the case of software, the transition of these activities is fluent (Preim and Dachselt 2015). A detailed design concept (e.g., in the form of prototypes) should be developed early in the process to receive feedback on usability. The final product can then be implemented according to the evaluated design concept. However, whether every dialogue is specified in depth before starting the implementation or whether for some elements only general requirements are defined can be decided individually for each project, and it also depends on the competencies of the developer (Preim and Dachselt 2015).

The following Figure 5 gives an example of how the specification, the design, and the implementation iteratively evolve as they are extended, refined, and improved based on formative evaluation results. These can provide new insights into the user's preferences and identify previously unrecognized risks (Janny and Pfeffer 2020). As formative evaluations do not have formal acceptance criteria and as the norm does not determine the number of design iterations, it needs to be decided for the specific project when to stop iterating or whether an iterative approach is used. For example, formative evaluations may end once there is no need for further improvement, when a certain quality level is achieved, or when the developer is confident that the summative evaluation will be successful. In general, the technical report of the norm recommends performing at least one formative evaluation before the summative evaluation.

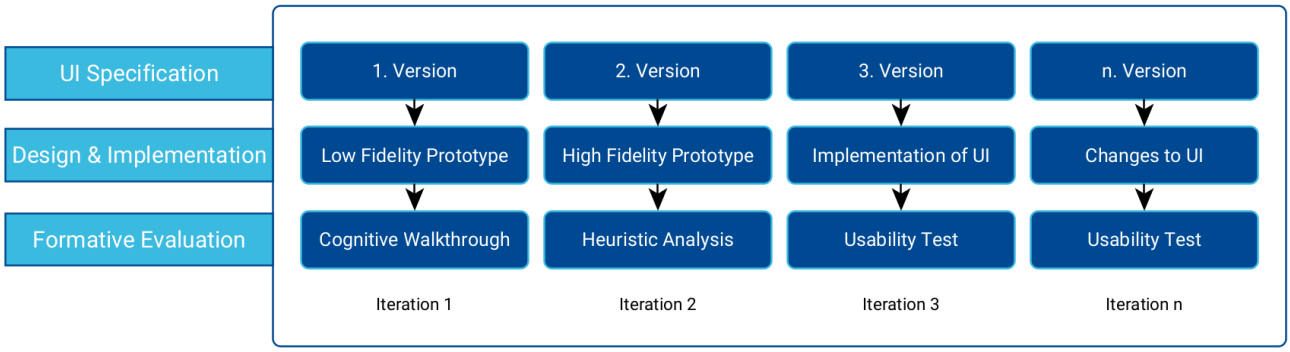

Figure 5. *Iterative Design and Development Process.*

4.5.1 User Interface Evaluation Planning

Usability evaluations form the centerpiece of the UE process and should be planned early in the process (Janny and Pfeffer 2020). Therefore, this planning activity can be performed with software development planning, which is the initial phase of the software lifecycle process (Johner, Hölzer-Klüpfel, and Wittorf 2021). The standard IEC 62366-1 requires establishing a user interface evaluation plan that explains the user interface specification (see Chapter 3.4.2.) will be evaluated. The plan is subdivided into formative and summative evaluation planning. The UI evaluation plan is a top-level plan that must always be updated throughout the software development process to reflect the current state of completed and planned evaluation activities. Having such a plan is very helpful, even in a research-based organization. It can help to synchronize the user interface development with corresponding evaluation activities. Nevertheless, the plan is allowed to vary in scope and complexity.

Evaluation methods must be chosen for formative and summative evaluation during the evaluation planning. The regulations do not dictate the evaluation methods or how many methods to use. In this respect, appropriate methods could be chosen depending on the complexity, novelty, and risk profile of the software to be developed (Janny and Pfeffer 2020).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Paz and Pow-Sang provide a comprehensive overview on several evaluation methods to choose from (Paz and Pow-Sang 2016). In general, evaluation methods can be differentiated into two types:

A. Analytical inspections methods: Usability experts verify a product against given requirements or specifications, e.g., heuristic analysis, cognitive walkthrough, expert review.

B. Test methods: Observation and inquiry of representative users, e.g., usability test, thinking aloud method.

Regardless of whether it is a formative or summative evaluation, if usability tests are employed, the IEC 62366-1 expects the following to be planned and documented according to Table 5. An important input for this planning is the use specification which determines the selection of representative test participants or test environments.

Table 5. *Planning of Usability Tests.*

Usability Test Planning
1. Which and how many representatives or intended users are involved, including the user profile they belong to. 2. Test environment and other conditions of use, e.g., Usability Lab, simulated or actual use environment. 3. Whether the accompanying documentation or training are provided during the test. 4. For summative evaluation: Method of collecting test data for conducting a subsequent analysis of the observed use errors, e.g., interview data, observational data.

Formative Evaluation Planning

Formative and summative evaluations differ regarding evaluation time. Formative evaluations are conducted iteratively throughout the UI design and development phase. These evaluations seek regular feedback from the design team on strengths, weaknesses, and the effectiveness of risk control measures on the current user interface design. Moreover, formative evaluations are required for the risk analysis, as they can reveal previously unrecognized use-related hazards and unanticipated use errors which result from design shortcomings. In general, the evaluation effort's focus, complexity, and formality can be chosen independently based on the need for additional understanding and clarification of user interactions and the need to assess multiple design options before the summative evaluation.

According to IEC 62366-1, the plan shall specify and document the following aspects for each formative evaluation, summarized in Table 6.

Table 6. *Contents of Formative Evaluation Planning.*

Formative Evaluation Planning
1. Objective of the evaluation 2. Chosen evaluation method 3. Evaluation focus, e.g., software version, which part of the user interface to be assessed. 4. Time plan that defines, when in the process the evaluation will be performed.

The following explanation shall assist the academic developer in choosing a suitable evaluation method. Since analytical inspection methods are associated with less effort but still allow the

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

identification of design flaws, they are recommended for early formative evaluations. In addition, formative usability tests can be conducted with little effort by just involving research group members. This is possible because, unlike summative evaluation, formative evaluations do not have to be done with representative users. Based on the cost-benefit analysis of user testing, Nielsen recommends distributing the testing budget on multiple but small usability tests involving three to five participants. This has the advantage that the same findings are observed in one test. However, after establishing a new design, another test can identify new usability problems. If the prospective SaMD has several highly distinct user groups, users of each user group should be included in testing. Showing the software product to nondevelopment-related persons regularly provides valuable insights into how users would use the product or prototype (Macaulay et al. 2009).

Moreover, design changes, in the beginning, are less expensive and time-consuming than in later stages as the software becomes more complex with increasing dependencies. Important to know is that formative evaluations, only as part of the review meeting in agile processes, will not be sufficient because it sets a testing focus on small features instead of overall interrelationships. Nevertheless, such a demonstration of the software to stakeholders to evaluate functionality is highly recommended.

Summative Evaluation Planning

On the other hand, a summative evaluation has to be performed on the final implementation of the user interface. Its purpose is to demonstrate that the tasks associated with hazard-related use scenarios can be conducted successfully and safely on the user interface by the intended users and within the intended use environment. This implies that potential use errors do not lead to unacceptable risks. However, it might not be feasible for academic organizations to simulate the use environment in a usability lab or conduct the testing in the actual use environment. Instead, academic organizations should rather focus on inviting representative users of each user group for the summative evaluation and then ask them to perform all tasks of the prior selected hazard-related use scenarios. Regarding the number of test participants, the technical report IEC 62366-2 recommends including 15 participants per distinct user group in a summative usability test. In case of minimal resources in the academic project, it can also be an option only to test the software in formative usability tests and then leave a more extensive summative evaluation to the industrial manufacturers.

Regarding the planning, the user interface evaluation plan for summative evaluation shall address the contents in Table 7. Summative evaluation planning might only be completed once the last formative evaluation is over.

Table 7. *Contents of Formative Evaluation Planning.*

Formative Evaluation Planning
1. Objective of the evaluation 2. Chosen evaluation method, including a rationale that objective evidence is produced by this method. 3. Which part of the user interface to be assessed. 4. If applicable, availability of accompanying documentation and provision of training. 5. Acceptance criteria

For a summative evaluation, the best evaluation method is a usability test involving representative users of all user profiles. With such a usability test with representative users, it would be easier to provide the required evidence for a safe user interface design that poses acceptable risks. Summative evaluations can require one or even multiple usability test sessions to assess all the

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

hazard-related use scenarios or to involve all user groups with distinct hazard-related use scenarios. Using other evaluation methods in summative evaluation is only justifiable if it is not practicable to simulate the use, and it is unethical to conduct the testing in actual use. Moreover, an expert review would be sufficient if the SaMD has no hazard-related use scenarios or not such ones that are associated with severe harm (IEC 62366-2 2015).

Furthermore, summative evaluation has formal acceptance criteria, which must be defined for the SaMD within the UI evaluation plan. Although the naming is identical, these acceptance criteria should be distinct from acceptance criteria that determine whether a user story or system requirement is fully implemented. Instead, meeting the acceptance criteria for summative evaluations implies that "the residual risks related to usability are controlled to acceptable levels". In contrast to other risks identified within the risk management process (ISO 14971 2019), use-related risks are usually not classified within a risk assessment matrix because the probability of use error occurrence cannot be determined analytically before the summative evaluation. Although empirical data to determine the probability of harm could be collected during summative usability testing, this will not provide reliable evidence if the testing was only done with a few participants. This is probably the case in many small academic projects (Johner, Hölzer-Klüpfel, and Wittorf 2021). Therefore, it is recommended to set acceptance criteria only considering the severity of harm. An example of acceptance criteria that determine when a summative evaluation was successful is as follows:

- No use error occurred that leads to harm or a specific severity level of harm.
- No new hazards, hazardous situations or hazard-related use scenarios were identified.

It is required that the criteria reflect the risk management policy for setting acceptance criteria, as defined in ISO 14971. Depending on the state of the technology and experience with similar software, such a policy determines how far the overall risk level must and can be reduced. In other words, the risk policy defines where to set the boundary between acceptable and unacceptable risks. So, regarding use-related risks, the criteria can either state that the occurrence of any use errors potentially causing harm is unacceptable or only in the case that use errors with a specific severity level occurred. This decision should correspond to the selection of hazard-related use scenarios based on severity.

4.5.2 User Interface Specification

For every software project, one of the first activities is eliciting requirements. The IEC 62366-1 itself does not provide any guidance on a systematic derivation of requirements. Instead, the software requirements analysis is defined in the software lifecycle norm IEC 62304 (IEC 62304 2015). The software requirements analysis identifies and translates user needs into stakeholder requirements (problem domain). Based on the stakeholder requirements, the norm demands deriving and documenting system requirements (solution domain) equivalent to software requirements in SaMD. System requirements formally specify software functions to fulfill the stakeholder demands (Geis and Johner 2015). Therefore, they describe, among other things, how the graphical user interface should behave during interactions with users in order to allow them to complete a certain task. The results of the software requirements analysis can be documented in the Software Requirements Specification (SRS). This document is typically structured into functional and non-functional requirements and constraints (Ebert 2022).

This brief description of the software requirements analysis determined by IEC 62304 was necessary for the usability engineering guideline to understand that IEC 62366-1 demands further functional and non-functional design requirements, specifically regarding usability aspects of the SaMDs' user interface. Developing a user interface specification builds the foundation for a user interface with good usability. The terminology "specification" might be confusing because the regulations, in reality,

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

expect a collection of so-called user interface requirements and not yet any concrete design concepts. Thus, the easiest way to create and document the UI specification would be to integrate the UI requirements into the SRS. The most important is to ensure that the UI specification considers the aspects described in the following.

UI Requirements derived from User Needs, Preferences and Capabilities

User interface requirements are defined by the norm as requirements that are specifically relevant to ensure safe use and good usability of the user interface. They specify user interface design characteristics from a black-box perspective. The IEC 62366-1 does not prescribe a unified manner to write UI requirements. UI requirements can be both functional as well as non-functional design requirements and are usually written using natural language notation. Most importantly is that UI requirements express the user's needs, capabilities and preferences which have been identified during earlier research on the intended users (e.g., what element design would support workflow most efficiently, preferred size / form) (Hastenteufel and Renaud 2019). Therefore, the UI specification should in particular be developed under consideration of the use specification. Especially the intended user group and the use environment place demands on the UI design. For example, the user interface might be viewed from different angles or by multiple people at the same time which affects the character size and placement of controls. Otherwise, the users might have some disabilities that should be considered in the design, such as a certain type of color-blindness, which places demands on the display color. In the following, some examples of UI requirements are provided:

- The display shall be visible at a distance of 1 m to three people standing side-by-side, with all being able to read the text."
- "When an on-going function requires the user to wait more than 3 s, the associated screen shall provide a progress indication."
- "Text shall be at least 14 point or larger to ensure legibility among individuals with less than normal visual acuity (e.g., users who are farsighted and might not be wearing their leading glasses)."

UI Requirements associated with the Implementation of Risk Control Measures

Minimizing risks related to usability is a major concern of usability engineering. All potentially harmful use errors and resulting hazardous situations – obtained from the risk analysis - should be, to the possible extent, controlled by reducing the severity of the harm or preventing the occurrence of a use error through optimization of the user interface design. Therefore, appropriate risk control measures must be determined, which guide users in using the software appropriately and protect them from exposure to serious risks (List, Ebert, and Albrecht 2017). Suitable risk control measures are determined during the design and development of the user interface. They usually imply further UI requirements that must be realized in the user interface design.

The selected measure for each use error that potentially results in a hazardous situation and leads to harm can be documented in the risk sheet or the UI specification, together with a unique ID, a description of the UI requirement, and the type of risk control (Geis and Johner 2015). In order to find the right risk control measure, it is essential to understand the root cause of a use error first, which might be an error in perception or cognition. Similar to risk control in the risk management process of ISO 14971, the usability norm requires to implement of at least one of the risk control measures, which are listed according to their priority in Table 8.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Table 8. *Types of Risk Control Measures, based on IEC and FDA standards (IEC 62366-1 2015; IEC 62366-2 2015; FDA 2016).*

Risk Control Measure Type	Description	Examples
Inherent safety by design	User interface is designed in a way that prevents the use errors	• System does not allow incorrect input or selection • Improve detectability or readability of controls • Remove features that can be mistakenly selected
Protective measure	Built-in protections against use errors as part of the user interface	• Dialogue that requires the user to actively confirm a critical action in order to proceed
Information for safety	Warnings or instructions on correct use that inform the user on potential risks	• Training Sessions • Online user manual (video, audio) • Highlight problematic inputs / results without requiring any confirmation

If possible, inherent safety by design should be the preferred measure because a redesign of the user interface will most likely reduce or remove the risk. However, this might only sometimes be possible. Therefore, the second option is protective measures. In the case of software, there are no protective measures in the actual sense, such as protective insulation. Instead, confirmation prompts can be implemented for critical actions that the user must actively confirm to proceed (Hastenteufel and Renaud 2019). Information for safety is the least preferred option because its effectiveness relies on whether the user has gained access to the information (e.g., training session might be needed but is not carried out), can learn from the information, and can recall the information when needed (FDA 2016). Moreover, for software products, the information for safety should be rather displayed on the user interface than in the accompanying documentation because it cannot be guaranteed that the user reads any additional material.

Requirements for the Accompanying Documentation and Training

The UI Specification shall also explain whether accompanying documentation or specific training is required to ensure the safe use of the SaMD. The accompanying documentation contains information on the SaMD for the user, e.g., instructions for use, a technical description, or an installation manual. It can be printed and auditory, visual, or multimedia files. Nevertheless, the primary goal should always be to develop a user interface that does not require any instructions because it is so obvious how to use the device. However, if any accompanying documentation is needed, then requirements for this document should be set which prevent risks resulting from incorrect, incomplete, overly complicated, or incomprehensible instructions for use (e.g., specify language/images/glossaries to be used, the targeting of different audiences, media, storage) (Johner, Hölzer-Klüpfel, and Wittorf 2021). As users often need more time to read complex user manuals, a rule of thumb is that the accompanying documentation should be written short and to the point, using code examples, illustrations, or video screencasts to get them started fast (List, Ebert, and Albrecht 2017).

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Furthermore, if training is selected as a risk control measure, additional training or training material requirements could be specified (e.g., training shall be embedded within the SaMD). In addition, the regulations require the manufacturer to implement a training capability by providing the required training materials and ensuring that the training is available to users during the expected service life of the SaMD. Ensuring this is not feasible for research organizations because post-production surveillance will happen after transferring the SaMD to the industry.

Design Principles, Heuristics and Style Guides

Although not required by the regulations, it is recommended in academia to determine which commonly known design principles, heuristics, and style guides should apply to the planned user interface. Using existing design guidelines can facilitate design decisions and ensure that the UI design ascribes to good UE practices as they embody proven practices for designing a usable and intuitive UI. Design guidelines can be differentiated into the following three types:

A. Design principles are ground rules that always apply, defined by norms (Geis and Johner 2015), e.g.

    a. The seven dialogue principles in ISO 9241-110 (Reuter and Kaufhold 2018)
    b. The principles of general information presentation in ISO 9241-112 (Hastenteufel and Renaud 2019)
    c. The principles on visual information in ISO 9241-125

B. Heuristics are rough rules of thumb, e.g.

    a. “Nielsen 10” (Nielsen 1994) - very similar to ISO 9241-110
    b. Schneiderman’s eight golden rules of interface design (Shneiderman and Plaisant 2010)
    c. Zhang’s fourteen principles for the health domain (Zhang et al. 2003) - suggested by the technical report IEC 62366-2

C. Style guides define common conventions for a specific platform that are generally known by users and thus do not leave room for interpretation, e.g.

    a. From Google, Microsoft, Apple
    b. Coloring rules for visualizations (Hattab, Rhyne, and Heider 2020)

4.5.3 Design and Implementation of the User Interface

Based on the UI requirements, in the design phase, a usability engineer should create detailed design concepts of the graphical user interface, which provide solutions for the required static and dynamic performance of the prospective user interface. Regarding this phase, the regulations are very fuzzy. They are only generally required to design and implement the user interface (IEC 62366-1 2015). The resulting design specifications can be a prototype or a text-based format. Because there are no regulatory demands regarding the design documentation, it can be decided freely for the specific academic project how to document design concepts in addition to the mandatory UI requirements. For example, a detailed design document summarizing general design decisions can be advantageous. It is essential always to ensure traceability between UI requirements, the detailed design concept, and the programming realization. The following will explain what activities and considerations are typically involved in the design process that starts with rough sketches and ends with detailed design specifications (Janny and Pfeffer 2020).

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

One major design decision is the choice of appropriate UI elements (e.g., checkboxes, radio buttons, text fields, slider, notifications) that implement the operating functions needed to support specific user actions (Geis and Johner 2015). Usually, a design draft also defines the placement of GUI elements, the navigation within the information space, the dialogue design, the visual configuration (e.g., fonts, colors, symbols), and the interaction design (Preim and Dachselt 2015). The task-based use scenarios determine the order in which UI elements must be placed to enable the intended workflow (Lipprandt and Röhrig 2018). In addition, available technologies and platforms decide whether specific design ideas are possible (Mentler 2018).

Within the iterative UI design process and before the implementation, user interface prototypes of different shapes and degrees of maturity are often used to specify and evaluate the user interface design and detect usability issues in early design phases (Geis and Johner 2015; Rosson and Carroll 2002). Whether creating prototypes before implementation is reasonable in academia can be decided individually for each project and is not mandatory. There are many methods for building user interface prototypes. A common way to characterize prototyping approaches is the differentiation between low-fidelity and high-fidelity prototypes. Fidelity describes the degree to which the prototype accurately represents the UI design, especially the interaction behavior. Low-fidelity prototypes create the first rough UI design (Hastenteufel and Renaud 2019). They mainly demonstrate the look of the user interface in sketchy, static screens, whereas high-fidelity prototypes also simulate the feel, meaning the functionality of a user interface. They reach from detailed drawings to fully interactive simulations. The target technology could also partially implement high-fidelity prototypes (Geis and Johner 2015).

The strengths and weaknesses of low- and high-fidelity prototypes, including the most common associated methods, are summarized in Table 9. The tool landscape for prototyping in user interface development is rapidly moving, and several tools are available on the market. Therefore, only some examples are in the following table.

Table 9. *Low- and High-fidelity Prototypes.*

	Low-fidelity prototype (abstract)	High-fidelity prototype (detailed)
Methods	● **Wireframes**: Schematics outline of the UI structure and areas occupied by informational content, e.g., wireframe.cc, Balsamiq ● **Paper mockups / prototypes:** Rough sketches of the UI design ● **Storyboarding**: Sequence of paper prototypes	● **Graphical mockups**: Images of the UI design and layout, e.g., Figma, bubble, Axure RP ● **HTML prototypes**: (Partly-) functional simulations using HTML ● **Interface builders**: Complete development environment for graphic design, e.g., Indigo Design
Strengths	● Less time & lower costs ● Evaluate multiple design concepts ● Identify requirements ● Narrow the design space	● Look & feel of the final product ● Partial / complete functionality ● Fully interactive ● “Living UI specification”
Weaknesses	● Poor detailed specification to code to ● Limited in modeling navigational and interaction flow ● Limited error checking	● Time-consuming & costly ● Inefficient for exploring the design space ● Blinds users to major representational flaws

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

When choosing a prototyping technique at different times in the process, the goals and resources of the project team, the audience, and how it will be presented must be considered (Rosson and Carroll 2002). Low-fidelity prototypes allow user interface designers to present their interpretation of UI requirements and explore design solutions at early development stages. By creating sketchy prototypes without programming skills, they can compare different design alternatives with little effort and then narrow down the design space.

Once the design space is narrowed to a few promising solutions, high-fidelity prototypes are required to assess the actual UI behavior. These realistic prototypes help to resolve detailed design decisions in layout, visual presentation, component selection, as well as testing of interactive features (Constantine 2003). Graphical mockups and HTML prototypes are as simple as low-fidelity prototypes regarding the time required for generation. They offer the advantage that clickable interfaces can be easily created to demonstrate the interactivity and a sense of the navigation flow. Finally, the most mature high-fidelity prototypes guide programmers in implementing the user interface in the run-time environment. In particular, graphical mockup screens are used by designers to facilitate the subsequent implementation and evaluation because they allow the creation of the layout and design close to reality.

When deploying the implemented software, it is recommended to use web-based services such as GitHub to host the software tool and enable long-term availability (List, Ebert, and Albrecht 2017). Moreover, a software product often depends on third-party software that must be installed in advance. Therefore, an easy-to-use installation interface that allows downloading and installing any required dependencies should be developed (Mangul et al. 2019). Alternatively, developers can wrap all software tools in a Docker container (List, Ebert, and Albrecht 2017) or package managers such as Bioconda. These package managers have the advantage of allowing the user to automate the installation, upgrades, and configurations of software tools (Grüning et al. 2018).

4.5.4 Formative Evaluation

Formative evaluations are an ongoing process during the design and development phase to evaluate the realization of the user interface specification. To gain insights from these evaluations, it is essential to analyze the evaluation results properly to decide whether design changes are necessary. The previous steps of the UE process would have to be repeated if:

1. Known usability problems preserve because risk control measures need to be more effective.

2. Unanticipated usability problems were discovered, including use errors, hazards, hazardous situations, or hazard-related use scenarios.

Revising the risk analysis and updating the UI specification is essential based on the evaluation results. Furthermore, after design modifications and risk control measures are realized, another formative evaluation should occur to ensure that no new risks arise from risk control measures, such as an error message that can be misunderstood (FDA 2016). In addition, if necessary, the User Interface Evaluation Planning has to be updated accordingly (IEC 62366-1 2015). The UI design is changed and evaluated until the developer believes all use-related risks are adequately controlled and there is no need for further refinement (IEC 62366-2 2015).

In addition, IEC 62366-1 requires documenting the results of formative evaluations, but the norm does not specify the content or structure of this documentation. The technical report of the norm states that a formative evaluation report does not have to be very extensive. Instead, it can be decided how much documentation is necessary to keep track of suggested improvements and existing usability issues (Geis and Johner 2015; List, Ebert, and Albrecht 2017). Thus, the

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

documentation template can be used to present the results in the formative evaluation report but is not mandatory.

As previously indicated, there are a multitude of usability evaluation techniques that could be applied in formative evaluations. For instance, systematic inspection of the user interface allows the identification of usability problems without the assistance of actual end-users (Branaghan 2018). Inspection is based on the review and assessment of the software interface. The effectiveness of such analytical methods is highly dependent on the extent to which hazards and use errors can be deduced analytically (FDA 2016). For inspection methods, the Johner Institute suggests a verification/falsification checklist, to document the evaluation results (Geis and Johner 2015). In a cognitive walkthrough, the focus is on testing compliance with functional UI requirements (verification). In contrast, in a heuristic analysis, the objective proof with a screenshot is only provided if a violation of one design guideline was detected (falsification). Based on the identified usability problems and suggested UI improvements, the developer can make decisions on UI adjustments.

Another option is to use a test method for formative evaluation. Usability tests are considered the gold standard of test methods (Janny and Pfeffer 2020). Formative usability tests allow us to discover unanticipated use errors only observed as intended users operate on the SaMD. They are an excellent way to explore whether the elements of the user interface are recognizable and understandable so that the user can perform the intended tasks accurately. Especially in the case of complex AI systems, transparency and traceability of the AI decision (explainability) and human understanding (causability) are essential to allow the human expert to understand and retrace the model's decisions (Holzinger et al. 2021). Therefore, usability testing is the best way to validate the extent to which the user can interpret the model (Beckers, Kwade, and Zanca 2021).

Usability tests can be performed on a running system or pre-mature prototypes of different fidelity (Branaghan 2018). The most convenient way would be a high-fidelity prototype, an early working version of the system. However, as waiting for running software might postpone usability testing into late development stages, even very rough low-fidelity prototypes can provide helpful input to redesign activities when involved in usability testing (Rosson and Carroll 2002). Usability tests on low-fidelity prototypes could be conducted by showing just a single screen and then verbally letting the test participants explain what actions they would take to pursue a specific goal (Hastenteufel and Renaud 2019). The documentation and analysis procedure suggested for summative evaluation (see Chapter 3.5.) can also be applied to these formative usability tests, but the extent can be short.

4.6 Summative Evaluation

Upon completion of the user interface implementation, a summative evaluation shall be performed to "obtain objective evidence that the user interface can be used safely" by evaluating the final user interface against the acceptance criteria defined in the UI evaluation plan (IEC 62366-1 2015). The regulations require that summative evaluation involves every selected hazard-related use scenario and that the intended users complete all associated tasks in a realistic simulation of actual use.

Therefore, if any accompanying documentation exists, it should be provided during the summative evaluation. Thereby, the user must receive training before the summative evaluation if this is a specified risk control measure.

In order to provide objective evidence that the risk is acceptable and that no further user interface improvements are necessary or practical, the results of the summative assessment must be analyzed and documented. If usability testing is the chosen evaluation method, then the testing procedure and the creation of a summative evaluation report should adhere to the structure suggested by Geis and Johner (Geis and Johner 2015):

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

1. Usability test protocol, including the test scenarios
2. Usability test report containing the collected data

Before conducting a usability test, test scenarios need to be written. Therefore, each hazard-related use scenario intended to be validated by the testing shall be translated into at least one test scenario. A test scenario shall ask the user to perform a task on the user interface and describe the context of this task without giving any hints on how to achieve it. Nielsen recommends writing realistic tasks and encouraging action, but leaving it open to the user how to use the software, instead of forcing them to interact with a specific feature. The prepared test scenarios will then be provided to the users during the usability test.

Furthermore, the present usability professionals should write test protocols to document which use errors they observed while the user was performing a task. Even though the user did not commit a use error but had some use difficulties, those problems should also be documented because they might indicate UI features needing improvement. Use difficulties are cases in which the test participant was confused or struggles to perform a task. If testing reveals that many users had the same difficulty, then the chance that they become a use error in the future increases (Wiklund, Kendler, and Strochlic 2016). During the test, the test facilitators should record further performance data such as successful task completion, use of error counts, and time to complete a task (Branaghan 2018). Measuring performance time is only necessary if the speed of user interaction is safety-related (FDA 2016).

Usability test protocol, including the test scenarios

Usability test report containing the collected data. In order to further understand the root cause of a use error, post-test interviews or questionnaires are a common method to let test participants comment on their behavior. This might even reveal that the use error was not caused by design but was instead a result of missing concentration or other problems in the specific test situation. Interview questions can also test whether the user perceived and processed the information provided in training or the instructions for use correctly. The final usability test report should then contain all the processed data on the aspects listed in Table 10 and a conclusion based on these results.

Table 10. *Collected Data in Usability Tests, own representation based on IEC 62366-2.*

Observational data	Interview data
• Successful task completion (& duration) • Use error counts • Description of observed use errors • New hazards, hazardous situations or hazard-related use scenarios	• Test participant's comments on their interactions with the SaMD • Test participant's reported root causes of their use errors and use difficulties

The test results will likely reveal some new or known use errors and difficulties during the summative evaluation. Each use error or use difficulty poses a new risk and has to be thoroughly analyzed in a subsequent risk analysis. At first, it has to be determined whether the use error was caused by the UI design and whether it can be linked to a hazardous situation that can lead to harm of a certain severity level (IEC 62366-2 2015). Under consideration of the acceptance criteria, it should then be decided for each use error whether the remaining risk can be tolerated or whether the resulting harm of a use error poses an unacceptable risk. The latter would indicate that the UI design should be "modified to reduce or eliminate the use problem and reduce the use-related risks to acceptable levels" (FDA 2016) unless the risk is "deemed to be acceptable concerning the benefit" of the SaMD. In order to prevent potentially harmful use errors, in the next step, the root cause of each use error

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

– that was classified to pose an unacceptable risk – must be identified by analyzing the observational and interview data from the summative evaluation. The root cause could be an error in perception or cognition that resulted from unfavorable UI design. This insight helps identify the parts of the user interface involved in the occurrence of the use error and should be the focus of further risk control measures. The previous steps of the UE process are then re-entered, as shown in Figure 6, to implement necessary design changes. The process is repeated if new hazards, hazardous situations, or hazard-related use scenarios have been discovered during the summative evaluation (IEC 62366-1 2015).

All in all, if any new usability problems are discovered or known problems persist, then the summative evaluation becomes a formative evaluation, and another final evaluation has to be performed afterward. In the second summative evaluation, the data obtained from the initial summative evaluation can be used for all parts that remain the same so that only the design changes are tested again. As soon as the acceptance criteria of the summative evaluation are met, it has to be decided whether further optimization of the user interface design is necessary and practicable to increase device safety. The previous process steps must be repeated if potential optimization remains to realize the design changes. After that, another summative evaluation has to be performed. Finally, a justification must be documented if further design improvement to reduce risk is not practicable. By finishing this step, the summative evaluation is completed. A successful summative evaluation implies adequate risk controls and use-related risks are controlled to acceptable levels. However, despite all the efforts to reduce risks, "it is practically impossible to make any device error-proof or risk-free; some residual risk will remain, even if best practices were followed in the design of the user interface" (FDA 2016).
Consequently, as the UE process ends, the summative evaluation results must be transferred to the risk management process. It is then the risk manager's responsibility to judge whether the overall residual risk of the SaMD is acceptable (Geis and Johner 2015). This final evaluation shall consider residual risks posed by software failures and those from use errors. Residual risk is the "risk remaining after risk control measures have been implemented" (BSI 2019). If enough persons were included in the summative evaluation, then the collected data on the occurrence of use errors could be used to estimate the probability of use-related risks. This would enable the risk manager to evaluate the reasonably foreseeable, use-related risks and the "normal" risks within a risk assessment matrix. However, determining the probability of use-related risks is often not possible even after the summative evaluation due to insufficient data and, thus, unreliable results (Johner, Hölzer-Klüpfel, and Wittorf 2021).

As indicated above, the risk management process determines when the SaMD is ready to be transferred to industrial manufacturers. After the software has been placed on the market, the MDR and the ISO 14971 demand to evaluate post-production surveillance data related to use errors to remove previously unidentified user interface shortcomings. However, these activities are not the responsibility of academic organizations and are not included in the usability engineering process.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

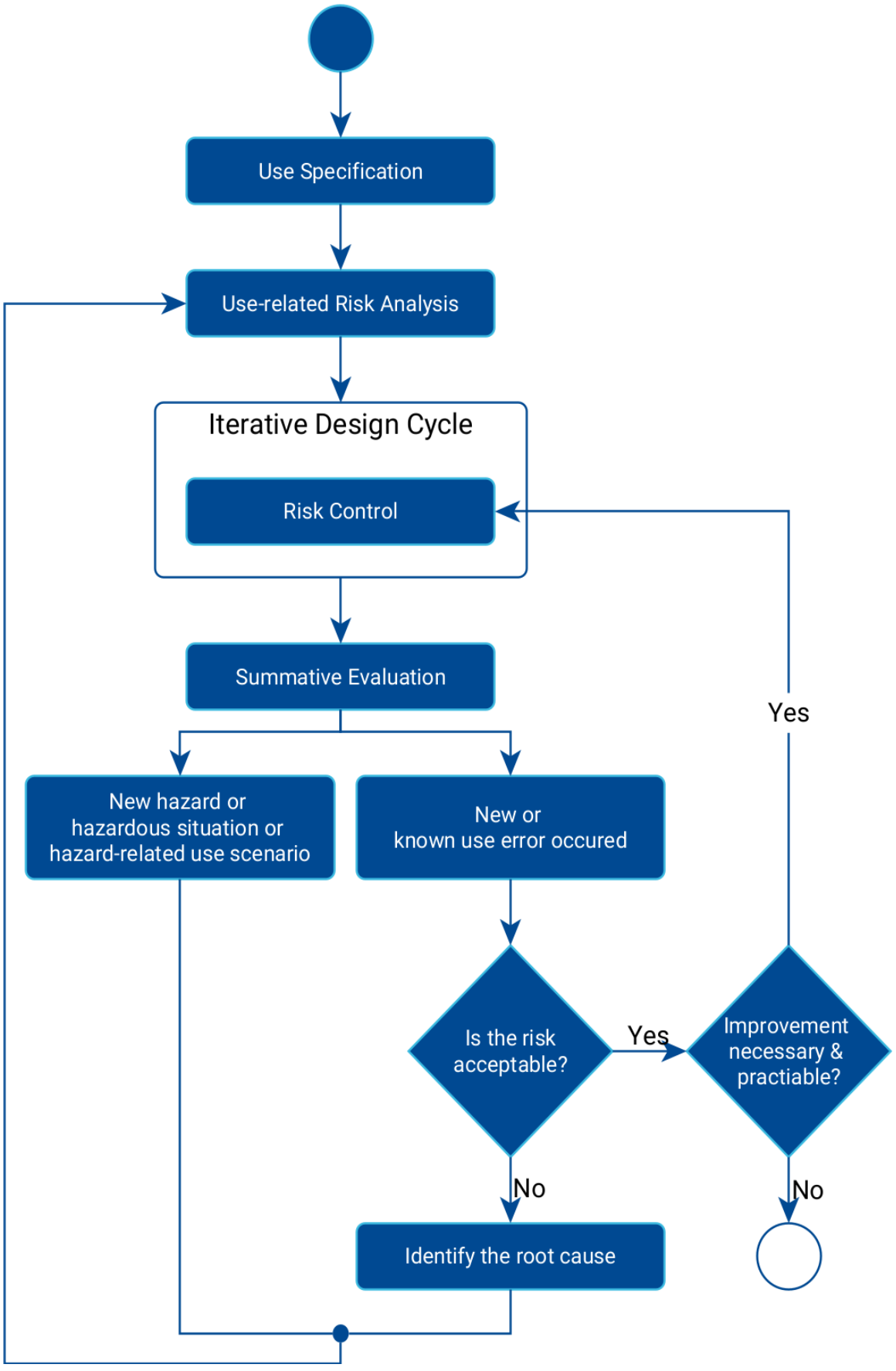

Figure 6. *Decision-making on Results of the Summative Evaluation.*

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

4.7 User Interface of Unknown Provenance

User Interface of Unknown Provenance (UOUP) is a user interface or a part of a user interface that was developed in the past, and that has no corresponding records of the UE process required by the IEC 62366-1 standard. This includes, for example, legacy software that has been developed and commercialized before the norm was published in 2015 or a user interface that was developed without the intention to be a medical device (Geis and Johner 2015).

The usability engineering process for such user interfaces or parts of user interfaces can be abbreviated. Thereby, the activities rely heavily on existing documentation and result in creating a usability engineering file. The following procedure only applies to unchanged parts of the UOUP, and changed parts must follow the whole usability engineering process:

1. Establish a use specification as described in Chapter 3.3.

2. Analyze and document available post-production data, e.g., complaints and field reports for incidents to identify use errors that could result in a hazardous situation.

3. Review the use-related risk analysis of the overall user interface, including the UOUP, and ensure that all hazards and hazardous situations have been discovered and documented. It is not necessary to write hazard-related use scenarios.

4. Verify and document that suitable risk control measures have been implemented for the new usability problems. If design changes are necessary, those parts are not considered UOUP anymore.

5. Based on the new information from steps 3 and 4, the overall residual risk has to be re-evaluated according to ISO 14971.

4.8 Usability Engineering Process Checklist

In addition to the usability engineering guideline proposed in the previous chapters, a concise checklist was developed to summarize the course of action derived from the IEC 62366-1 and adjusted to academic capabilities. The described vital steps will make it easier for academic researchers to understand and establish a UE process.

1. Describe the context of the use of the software with a focus on those aspects that influence its usability. This shall most notably include the intended medical indication, patient population characteristics, user profile, and use environment.

2. Determine how to identify use-related user interface risks and decide where to document the results. The goal is to describe hazard-related use scenarios with all their tasks and sequences. Therefore, examine the following components:
    a. Identify safety-related user interface characteristics by determining those operating functions that have an impact on the safety of the software.
    b. Identify potential use errors that users might conduct when interacting with the UI.
    c. Identify foreseeable hazards and hazardous situations that might result from use errors.
    d. Adapt the categorization scheme defined in the risk management process for measuring the severity of the associated harm.

3. Select hazard-related use scenarios that should be included in summative evaluation based on the severity of harm and a predefined selection scheme.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

4. Establish a plan for formative and summative evaluation activities on the user interface and determine the required content and detail of the planning document. Set acceptance criteria for a successful summative evaluation.

5. Derive and document UI requirements based on insights from user research, risk analysis, and formative evaluations. UI requirements include functional and non-functional design requirements and can reflect the following aspects:
    a. User needs, preferences, and capabilities influence the UI design.
    b. Risk control measures place demands on the UI design to mitigate use-related risks.
    c. (If applicable) demands for the accompanying documentation and training materials.

6. Optional: Collect a list of design principles, heuristics, or style guides that ensure the recognizability and interpretability of relevant information (not required by the IEC 62366-1, but they can aid developers in designing a user interface with good usability).

7. Design and implement the user interface according to the UI requirements. Thereby, prototyping and creating a detailed design document summarizing general design decisions can be advantageous.

8. Perform formative evaluations during the development process to the extent possible and reasonable. Determine which methods are suitable to evaluate the effectiveness of specific risk control measures and reveal unanticipated usability problems. Document as much as necessary to keep track of unresolved design issues and planned design improvements.

9. Perform a summative evaluation to provide objective evidence that the intended users can complete the tasks successfully and safely on the user interface. Analyze the usability test results to determine whether further refinement is necessary and practicable.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

5 Conclusion

This guideline proposes a usability engineering guideline that can assist research organizations in establishing a tailored UE process for developing software intended for medical use. A UE process is a design and development process for the user interface. Being aware that producing regulatory standards-compliant software is not feasible for most research organizations, the guideline balances effort and regulatory compliance by confining the requirements of the IEC 62366-1. The final UE guideline emphasizes and describes key process steps, provides concrete explanations for abstract demands of the norm, and gives recommendations for documenting the UE activities. The proposed guideline is centered on procedures for the software's use specification, use-related risk analysis, UI evaluation planning, UI specification, design and implementation of the user interface, continuous formative evaluations, the final summative evaluation, and an abbreviated workflow for UOUP. However, depending on the specific circumstances and needs, the research organization or group can deliberately choose a set of UE activities that can deviate from what was suggested within this UE guideline. Nevertheless, the guideline provides a starting point for establishing a limited usability engineering process and thus reduces the hurdle in academia to consider and adhere to regulatory demands. Ultimately, documenting the UE activities ensures a smooth project handover to other research groups and thus supports medical device manufacturers in preparing academic software products for the clinical certification procedure. In summary, implementing presented processes will prime academic software for clinical admission, which greatly accelerates and facilitates the technology transfer to commercial manufacturing. Moreover, following a usability engineering process in academic software development will increase the usability of the final product and thus reduce use errors with potentially harmful consequences for patients.

Regarding the use of the guideline, it should be noted that this guideline does not provide a shortcut or simplify the development of regulatory standards-compliant medical software. In the case of an intended formal admission and certification as SaMD, strict compliance with the regulations of national notified bodies is indispensable. Instead, as mentioned before, the guideline intends to reduce the hindering factors that keep academic organizations from producing compliant software according to the IEC 62366-1 by suggesting a limited UE process that keeps the effort in a range that most research organizations can handle. Furthermore, it should be considered that the usability engineering process is just one component of the overall software development process. Therefore, only considering the demands of the usability norm is not sufficient to produce compliant software. In addition, a quality management system (ISO 13485), a risk management process (ISO 14971), and a software lifecycle process (IEC 62304) must be established.

As already indicated, the developed guideline is mostly based on the international standard IEC 62366-1. This norm incorporates the regulatory requirements of several different countries, including the EU, and thus applies across national borders. However, the regulatory requirements for a successful SaMD admission can differ to some extent in other nations. For example, the FDA, which is the responsible institution for SaMD admission in the United States of America, officially recognizes the IEC 62366-1. They define further requirements beyond the scope of the international standard, which are not considered in the guideline. Consequently, when targeting markets outside the EU, it is necessary to examine whether the particular national requirements place further demands on medical device development.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

6 References

Barredo Arrieta, Alejandro, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, et al. 2020. “Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI.” *Information Fusion* 58 (June): 82–115. https://doi.org/10.1016/j.inffus.2019.12.012.

Beckers, R., Z. Kwade, and F. Zanca. 2021. “The EU Medical Device Regulation: Implications for Artificial Intelligence-Based Medical Device Software in Medical Physics.” *Physica Medica: European Journal of Medical Physics* 83 (March): 1–8. https://doi.org/10.1016/j.ejmp.2021.02.011.

Branaghan, Russell J. 2018. “Human Factors in Medical Device Design: Methods, Principles, and Guidelines.” *Critical Care Nursing Clinics of North America*, Human Factors and Technology in the ICU, 30 (2): 225–36. https://doi.org/10.1016/j.cnc.2018.02.005.

BSI. 2019. “Guide to the Development and Inclusion of Aspects of Safety in International Standards for Medical Devices.” PD ISO/IEC GUIDE 63:2019. BSI British Standards. https://doi.org/10.3403/30390373.

Butler, Keith A. 1996. “Usability Engineering Turns 10.” *Interactions* 3 (1): 58–75. https://doi.org/10.1145/223500.223513.

Constantine, Larry L. 2003. “Canonical Abstract Prototypes for Abstract Visual and Interaction Design.” In *Interactive Systems. Design, Specification, and Verification*, edited by Joaquim A. Jorge, Nuno Jardim Nunes, and João Falcão E Cunha, 2844:1–15. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-39929-2_1.

Ebert, Christof. 2022. *Systematisches Requirements Engineering: Anforderungen ermitteln, dokumentieren, analysieren und verwalten*. 7., Überarbeitete und aktualisierte Auflage. Heidelberg: dpunkt Verlag.

Elahi, Bijan. 2018. *Safety Risk Management for Medical Devices*. London: Academic Press, an imprint of Elsevier.

FDA. 2016. “Applying Human Factors and Usability Engineering to Medical Devices: Guidance for Industry and Food and Drug Administration Staff.” Standard FDA-2011-D-0469. Center for Devices and Radiological Health: Food and Drug Administration. https://www.fda.gov/regulatory-information/search-fda-guidance-documents/applying-human-factors-and-usability-engineering-medical-devices.

Geis, Thomas, and Christian Johner. 2015. *Usability Engineering als Erfolgsfaktor: effizient IEC 62366- und FDA-konform dokumentieren*. 1. Aufl. Praxis Medizintechnik. Berlin Wien Zürich: Beuth.

Grüning, Björn, Ryan Dale, Andreas Sjödin, Brad A. Chapman, Jillian Rowe, Christopher H. Tomkins-Tinch, Renan Valieris, and Johannes Köster. 2018. “Bioconda: Sustainable and Comprehensive Software Distribution for the Life Sciences.” *Nature Methods* 15 (7): 475–76. https://doi.org/10.1038/s41592-018-0046-7.

Hastenteufel, Mark, and Sina Renaud. 2019. *Software als Medizinprodukt: Entwicklung und Zulassung von Software in der Medizintechnik*. Wiesbaden: Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-26488-8.

Hattab, Georges, Theresa-Marie Rhyne, and Dominik Heider. 2020. “Ten Simple Rules to Colorize Biological Data Visualization.” *PLOS Computational Biology* 16 (10): e1008259. https://doi.org/10.1371/journal.pcbi.1008259.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

Holzinger, Andreas, Bernd Malle, Anna Saranti, and Bastian Pfeifer. 2021. “Towards Multi-Modal Causability with Graph Neural Networks Enabling Information Fusion for Explainable AI.” *Information Fusion* 71 (July): 28–37. https://doi.org/10.1016/j.inffus.2021.01.008.

IEC 62304. 2015. “Medical Device Software - Software Life Cycle Processes - Amendment 1.” Standard IEC 62304:2006/Amd 1:2015. Geneva, CH: International Organization for Standardization. https://www.iso.org/standard/64686.html.

IEC 62366-1. 2015. “Medical Devices - Part 1: Application of Usability Engineering to Medical Devices.” Standard IEC 62366-1:2015. Geneva, CH: International Organization for Standardization. https://www.iso.org/standard/63179.html.

IEC 62366-2. 2015. “Medical Devices - Part 2: Guidance on the Application of Usability Engineering to Medical Devices.” Standard IEC TR 62366-2:2015. Geneva, CH: International Organization for Standardization. https://www.iso.org/standard/63179.html.

ISO. 2019. “ISO 14971:2019.” *International Organization for Standardization*. https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/07/27/72704.html.

ISO 9241-11. 2018. “Ergonomics of Human-System Interaction - Part 11: Usability: Definitions and Concepts.” Standard ISO 9241-11:2018. Geneva, CH: International Organization for Standardization. https://www.iso.org/standard/63500.html.

ISO 14971. 2019. “Medical Devices - Application of Risk Management to Medical Devices.” Standard ISO 14971:2019. Geneva, CH: International Organization for Standardization. https://www.iso.org/standard/72704.html.

Janny, Benedikt, and Stefan Pfeffer. 2020. “Formative und summative Usability Evaluationen medizintechnischer Produkte - Menschzentriert entwickeln und dabei die regulatorischen Anforderungen erfüllen.” https://doi.org/10.18420/muc2020-up-0142.

Johner, Christian, Matthias Hölzer-Klüpfel, and Sven Wittorf. 2021. *Basiswissen medizinische Software: Aus- und Weiterbildung zum Certified Professional for Medical Software*. 3., Überarbeitete und aktualisierte Auflage. Heidelberg: dpunkt.verlag.

Lipprandt, Myriam, and Rainer Röhrig. 2018. “Sicherheitskritische Mensch-Maschine-Interaktion in der Medizin.” In *Sicherheitskritische Mensch-Computer-Interaktion: Interaktive Technologien und Soziale Medien im Krisen- und Sicherheitsmanagement*, edited by Christian Reuter, 297–315. Wiesbaden: Springer Fachmedien. https://doi.org/10.1007/978-3-658-19523-6_15.

List, Markus, Peter Ebert, and Felipe Albrecht. 2017. “Ten Simple Rules for Developing Usable Software in Computational Biology.” *PLOS Computational Biology* 13 (1): e1005265. https://doi.org/10.1371/journal.pcbi.1005265.

Macaulay, Catriona, David Sloan, Xinyi Jiang, Paula Forbes, Scott Loynton, Jason R. Swedlow, and Peter Gregor. 2009. “Usability and User-Centered Design in Scientific Software Development.” *IEEE Software* 26 (1): 96–102. https://doi.org/10.1109/MS.2009.27.

Mangul, Serghei, Lana S. Martin, Eleazar Eskin, and Ran Blekhman. 2019. “Improving the Usability and Archival Stability of Bioinformatics Software.” *Genome Biology* 20 (1): 47. https://doi.org/10.1186/s13059-019-1649-8.

MDCG. 2019. “MDCG 2019-11 Guidance on Qualification and Classification of Software in Regulation (EU) 2017/745 – MDR and Regulation (EU) 2017/746 – IVDR,” November. https://ec.europa.eu/docsroom/documents/37581.

Mentler, Tilo. 2018. “Usability Engineering und User Experience Design sicherheitskritischer Systeme.” In *Sicherheitskritische Mensch-Computer-Interaktion: Interaktive Technologien und Soziale Medien im Krisen- und Sicherheitsmanagement*, edited by Christian Reuter, 41–60. Wiesbaden: Springer Fachmedien. https://doi.org/10.1007/978-3-658-19523-6_3.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

Nielsen, Jakob. 1994. *Usability Engineering*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Paz, Freddy, and José Antonio Pow-Sang. 2016. “A Systematic Mapping Review of Usability Evaluation Methods for Software Development Process.” *International Journal of Software Engineering and Its Applications* 10 (1): 165–78. https://doi.org/10.14257/ijseia.2016.10.1.16.

Preim, Bernhard, and Raimund Dachselt. 2015. *Interaktive Systeme: Band 2: User Interface Engineering, 3D-Interaktion, Natural User Interfaces*. eXamen.press. Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-642-45247-5.

Ravizza, Alice, Andres Diaz Lantada, Luis Ignacio Ballesteros Sánchez, Federico Sternini, and Cristina Bignardi. 2023. “Techniques for Usability Risk Assessment during Medical Device Design.” In , 207–14. https://www.scitepress.org/Link.aspx?doi=10.5220/0007483102070214.

Reuter, Christian, and Marc-André Kaufhold. 2018. “Usable Safety Engineering sicherheitskritischer interaktiver Systeme.” In *Sicherheitskritische Mensch-Computer-Interaktion: Interaktive Technologien und Soziale Medien im Krisen- und Sicherheitsmanagement*, edited by Christian Reuter, 17–40. Wiesbaden: Springer Fachmedien. https://doi.org/10.1007/978-3-658-19523-6_2.

Riemenschneider, Mona, Joachim Wienbeck, André Scherag, and Dominik Heider. 2018. “Data Science for Molecular Diagnostics Applications: From Academia to Clinic to Industry.” *Systems Medicine* 1 (1): 13–17. https://doi.org/10.1089/sysm.2018.0002.

Rosson, Mary Beth, and John M. Carroll. 2002. *Usability Engineering: Scenario-Based Development of Human-Computer Interaction*. 1st ed. Morgan Kaufmann Series in Interactive Technologies. San Francisco: Academic Press.

Shaheen, Momina, Tayyaba Anees, Muhammad Junaid Anjum, and Aimen Anum. 2021. “Usability Engineering Process for Medical Devices.” In *Proceedings of the Future Technologies Conference (FTC) 2020, Volume 3*, edited by Kohei Arai, Supriya Kapoor, and Rahul Bhatia, 365–81. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing. https://doi.org/10.1007/978-3-030-63092-8_25.

Shneiderman, Ben, and Catherine Plaisant. 2010. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. 5th ed. Boston: Addison-Wesley.

Wiklund, Michael E., Jonathan Kendler, and Allison Y. Strochlic. 2016. *Usability Testing of Medical Devices*. Second edition. Boca Raton, FL: CRC Press, Taylor & Francis Group.

Wood, Larry E., ed. 1998. *User Interface Design: Bridging the Gap from User Requirements to Design*. Boca Raton: CRC Press.

Zhang, Jiajie, Todd R Johnson, Vimla L Patel, Danielle L Paige, and Tate Kubose. 2003. “Using Usability Heuristics to Evaluate Patient Safety of Medical Devices.” *Journal of Biomedical Informatics*, Patient Safety, 36 (1): 23–30. https://doi.org/10.1016/S1532-0464(03)00060-1.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.