Ensuring Quality

The Key Elements of Technical Documentation for Medical Devices

Introduction

ImFusion develops advanced medical image processing software which can be used by industry customers as part of a certified medical device. While ImFusion only provides software, industry customers typically build a product which combines software and hardware. In order to place a product on the market as medical device, no matter if it's an apparatus, a device or a stand-alone software, the manufacturer must submit a set of documents to the regulatory authorities. This set of documents is called technical documentation or technical file. The technical documentation must contain a general description of the product, the design specifications, the results of the risk analysis, a description of verification techniques and the (pre-)clinical evaluation. In the field of software as a medical device, the software can be a stand-alone software, an integral part of a medical device or a software used in the manufacture or maintenance of a medical device. To support its customers in the certification process ImFusion can deliver the technical documentation for its software components as needed for certification as a medical device. Depending on the kind of project this can either be a full set of documentation for a custom piece of software according to IEC 62304 or a set of documents to validate existing ImFusion components as so called SOUP. In addition, ImFusion has been certified according to EN ISO 13485 since 2016 which makes it easy for medical device manufactures to approve it as a critical supplier.

In the following we will explain the typical workflow for developing and documenting customized software for use in a medical device.

Software Development Process

The software development process consists of several stages ranging from requirement elicitation to post-release maintenance. ImFusion follows a software development process for medical software. This process is based on the standard for risk management (EN ISO 14971), the one for software development (IEC 62304) and the one for usability (IEC 62366). The software lifecycle management process implemented by ImFusion is structured in successive steps that integrate the requirements of EN ISO 13485. Each step can be assigned to a specific phase:

- Planning Phase,
- Design Phase,
- Implementation Phase and
- Verification Phase.

The development process aims at conceiving a software system that addresses user requirements. Based on these user requirements we plan the software development and write software requirements. After that, the individual software features are implemented and verified by unit testing. Next, the software features are integrated and verified by use case testing and integration tests. Finally, user requirements are validated during system validation. The output of this process is a fully tested and working software system along with all development and product documentation. All related documents are assigned to one specific phase each. The process is iterative and incremental.

Usually, the system validation, including field testing, usability testing and acceptance tests, is conducted by the manufacturer who places the medical device on the market. Once the software has been released the maintenance phase begins. This includes fixing of bugs and other issues that occur during use in the field.

During the development process two additional processes are running in parallel. These are the risk management and the configuration management process, since these might be impacted by the steps of the development process. Risk Management is an overall process that is performed at the system level. Nevertheless, risks resulting from software design or associated with the use of the software are specifically identified, evaluated and mitigated. All risk

management activities follow EN ISO 14971 and IEC 62304 and are documented in the Product Risk Analysis Matrix.

The software configuration management sub-process ensures the developed code and the development environment are controlled. This means that software components, software tools and the software itself are managed in terms of changes, versions and that, when necessary, they are verified prior to use. Each software release is assigned a version in order to correctly identify the stage of development and to correctly trace changes.

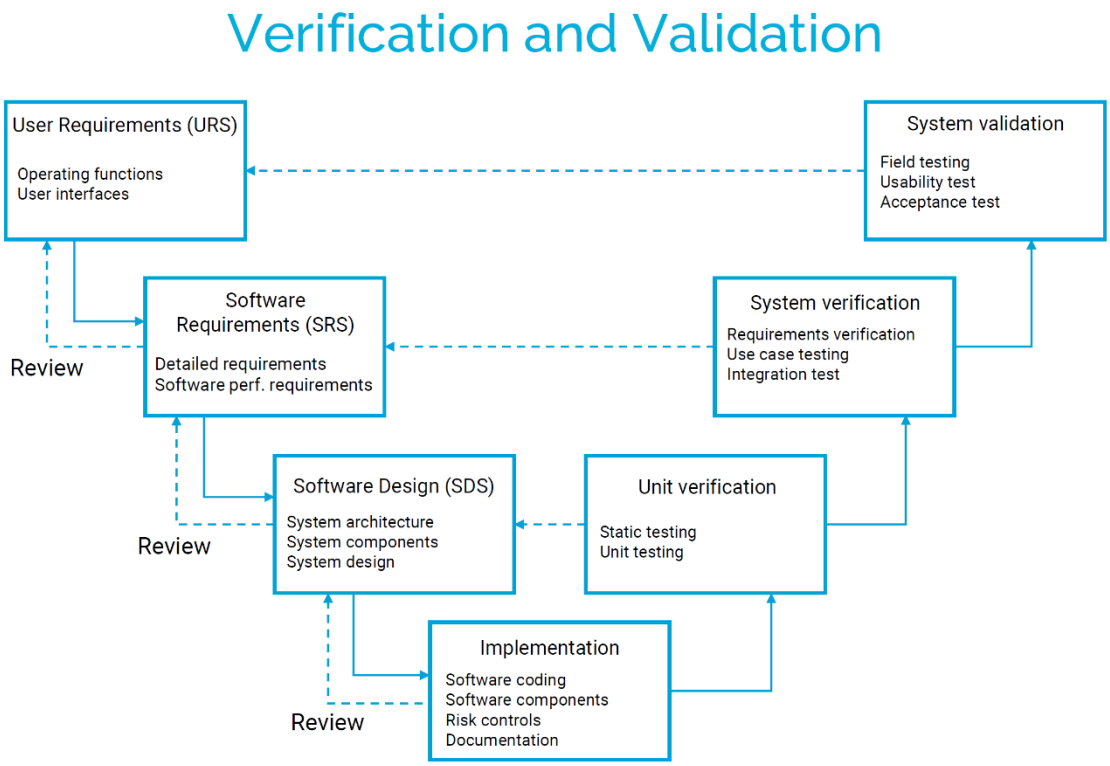

Figure 1 The V-model reflects the relationship between development phases and corresponding testing activities.

Technical Documentation

As a supplier for manufacturers of medical devices we provide the technical documentation for the features we are developing. All documents are under version control and an approval process ensures that only documents which have been reviewed and approved are released.

User Requirement and Software Requirement Specifications, test cases and risk items are tracked in an issue management system and links are establish to facilitate the creation of a traceability report.

All code is tracked with a version control system and documented.

In the following sections the documents created during development are described in more detail.

Planning Phase

The planning phase is a critical initial step in the development and regulatory process. During this phase, the focus is on defining the overall strategy and framework for the product. Key activities in this phase include evaluating applicable regulatory requirements and standards, risk assessment, defining User Requirements and creating the Software Development Plan.

The Planning Phase consists of the following documents:

- User Requirement Specification
- Software Development Plan
- Planning Review Report

User Requirement Specification (URS)

At the beginning the customer provide the product requirements (user requirements). These User Requirement Specifications (URS) are used to plan the software. URS contain the specifications of a software from the user's point of view and are typically non-technical and high-level. These items could contain functional, usability, performance, safety, interface and environment

requirements. ImFusion adds the URS as “User Requirement” issues to an issue tracking system. The customer receives an export of the URS as part of the technical documentation.

Software Development Plan

The Software Development Plan (SDP) describes the life-cycle for the software development project. Here, the phases of the development, the activities carried out in each phase and the deliverables to be produced are described. This covers the setup of the development project, the initial specification, design, implementation and test until the software is released. Also, the software risk management activities during those phases are defined. It is created during the planning phase of the software development and reflects the project plan.

The SDP includes the safety classification for the product, which affects the risks analysis and the qualification of SOUPs and tools. Additionally, the coding standards are described.

The software development life-cycle contains all phases from the planning of the software to decommissioning. Depending on the scope of the order some of the steps are performed by ImFusion while others are performed by the customer. These responsibilities and roles are defined in the SDP.

Furthermore, in the SDP software unit criteria and criteria for the code reviews are defined.

Planning Review Report

The Planning Phase is closed with the Planning Review Report. It contains a review checklist with which the completeness of the documents of the Planning Phase is checked. Additionally, it is checked whether the URS are entered correctly and whether they are free from contradiction or ambiguity. Also the fulfilment of regulatory requirements is checked.

Design Phase

The design phase is a pivotal stage where the conceptualization and planning from the previous phase come to life. During this phase, the focus is on transforming the defined requirements and concepts into a functional, safe, and effective software product. Key activities in the design phase include architecture design, detailed design, risk mitigation and software development.

The Design Phase consists of the following documents:

- Software Requirement Specification
- Software Architecture Documentation
- Detailed Design Documentation
- Tool Validation
- Software Risk Analysis
- Cybersecurity Risk Analysis
- Design Review Report

Software Requirement Specifications (SRS)

On the basis of the URS the Software Requirement Specifications (SRS) are written. SRS describe the software specifications on a more detailed and technical level. This can also include requirements on the platform (e.g. hardware, databases, interfaces).

ImFusion adds the SRS as “Software Requirement” issues to an issue tracking system. The SRS are linked to the URS they are deriving from, to maintain traceability. The customer receives an export of the SRS as part of the technical documentation.

For every Software Requirement Specification test cases are written. ImFusion adds the test cases as test issues to its issue tracking system.

Risk control measures are also documented as SRS. These are marked as “Risk Control” issues and referenced in the Software Risk Analysis.

Software Architecture Documentation

The Software Architecture Documentation (SAD) gives an overview of the system context, the hardware environment, the software systems and items and the system behavior. The system context is described by a system context diagram giving an overview of the high-level components. In the description of the hardware environment the needed hardware components are stated. Additionally, the software safety classification for the system and its related components is specified in the SAD.

All implemented components and software items are described. Next to the description of the technical functionality, for each software item all related SRS contributing to or mitigating risks are listed. The architecture of the single software items is described by a system context diagram. In this diagram the relationships and interfaces between the involved components and modules are shown. Additionally, for each software item the dependencies to e.g. SOUPs or other software items, as well as the required resources and important data types are described.

In the system behavior section the start-up of the ImFusion framework, the communication of the customer application with the ImFusion framework and the synchronization of the system are described.

All code written by ImFusion follows specific Coding Guidelines. Additional applicable guidelines (e.g. design constraints, error handling or design patterns) are stated in the SAD.

Detailed Design Documentation

The Detailed Design Documentation (DDD) consists of code-level documentation.

Tool Validation

All tools that are used during software development need to be validated. For each tool a Tool Validation Plan is created that covers the tool-specific requirements and functions. The validation plan contains a tool description that states the type and purpose of the tool including a high-level overview of the functionality used. Furthermore, the environment of the tool is given, e.g. if it is

running on a server or a local network, if backups are performed, if it is a web application or an installable program and if a bug tracker is available.

Additionally, a risk analysis is performed listing all hazards that might occur during use. These hazards are controlled and tested by tool-specific validation steps and verification measures that are conducted whenever a new tool or a new version of a tool is used.

The result of the tool validations is summarized in a Tool Validation Report giving an overview of all used tools with vendor, version, validation date and a reference to the latest validation report.

Software Risk Analysis

The Software Risk Analysis (SRA) is a very important part of the software development process and the technical documentation. The risk management process runs in parallel to the software development process, since changes during the development may have an impact on risks deriving from functions of the software. According to the use case and the features of the product, hazards and risks are identified and updated during all phases of the product life cycle and shall be reduced to a minimum.

We always consider the current state of the art during the development or design changes of products. Each potential risk shall be evaluated and addressed appropriately either by a design modification or by a risk control measure in software or, preferably, in hardware. When assessing the risk, the main variables are probability of occurrence and severity of harm.

During Hazard Analysis, the development team ensures that at least the following potential causes of software items contributing to hazardous situations are considered:

- Incorrect or incomplete specification of functionality
- Software defects in the functionality
- Failure or unexpected results from SOUPs
- Hardware failures or other software defects that could result in unpredictable software operation
- Reasonably foreseeable misuse by a user

For risk mitigation the following measures shall be used:

- Elimination / Minimization by technical measures
- Implementation of reasonable protective measures including alarm signals against hazards/risk that cannot be eliminated by technical measures.
- Briefing the user about hazards/risk that could not be reduced sufficiently by the previous two measures. This measure has no minimizing effect on the risk estimation.

A lower method shall be used only if a superior method is exhausted. Multiple methods of risk mitigation can be combined.

The SRA lists and classifies all hazardous situations identified in the system risk analysis. According to the system risk analysis, the following risk classes are used:

- Minor: malfunction unlikely to result in any injury
- Moderate: malfunction directly results in minor injury to patient and/or operator
- Major: malfunction directly results in life threatening or serious injury to patient and/or operator

If needed these risk classes can be adjusted so that they fit a device specific risk matrix.

The SRA describes the contributing SRS, the reasons/error sources as well as the implemented software risk mitigation measures. The goal is to minimize the risk as much as possible. If a risk control measure is implemented in software, this risk control measure has to be evaluated to identify and document any new sequences of events that could result in a new hazardous situation.

The software items and units associated with each SRS are listed in the software architecture documentation in the sections relating to contributing risk and mitigating risk. The SRA does not include usability risks, which are assessed and managed by the customer. All risks are documented in an issue management system and are linked to the contributing and mitigating SRS. This relationship can be tracked in the Traceability Report, as well.

Cybersecurity Risk Analysis

Next to the risks related to patient safety we conduct, if needed, a Cybersecurity Risk Analysis. Here, assets, threats, and vulnerabilities and their impact on device functionality and end users/patients are defined. The risk analysis can also consider data security topics. The likelihood and risk levels is assigned to every threat and vulnerability. To reduce likelihood or risk level of the risks suitable mitigation strategies are implemented. Afterwards, the residual risk is determined. If necessary according to the defined risk acceptance criteria further mitigations are implemented.

Design Review Report

The Design Phase is closed with the Design Review Report. It contains a review checklist with which the completeness of the documents of the Design Phase is checked. Additionally, it is checked whether the URS are completely covered by SRS and whether these SRS correctly implement the requirements from the URS and are free from contradiction or ambiguity. Furthermore, the risk analysis is reviewed for completeness and correctness. If needed, the Software Development Plan might be updated during the Design Phase. Therefore, the validity of the Software Development Plan is checked.

Implementation Phase

The implementation phase is the stage where the designed software is developed, which involves the actual coding, integration of components, and finalizing the software for real-world use. Key activities in the implementation phase include coding and development, component integration and verification activities, the latter including software unit verification to ensure each implemented unit behaves as expected. Unit testing is performed during the development process on a regular basis to verify that the implemented functionalities are those that have been specified and that the components used behave as expected. This level of testing is performed in the development environment.

The Implementation Phase consists of the following documents:

- Code Review Report
- Test Code Review Report
- Traceability Report
- SOUP Qualification
- Implementation Review Report

Code Review and Test Code Review Report

During the Code Review and Test Code Review the written code is checked for plausibility, completeness and correct implementation. The reviewers check whether the code reflects the requirements properly and whether all risk control measures are covered. Additionally, it is checked whether the code is free from contradictions with the interfaces documented in the detailed design of the software unit. All code that is written at ImFusion follows our internal coding guidelines. The fulfilment of these coding guidelines is verified during the code review. In addition to the coding guidelines the reviewers check the correct implementation of error handling, memory management and thread safety and whether all variables are initialized when they are defined. Furthermore, the general readability of the code and the test coverage is reviewed.

Traceability Report

In order to ensure that the developed software addresses the Software Requirements Specification and that each requirement has been properly verified, traceability between system requirements, software requirements, software system tests, and risk control measures implemented in software is established. We use our issue tracking system to enable the traceability between requirements, specification, risk mitigation measures and verification and validation tests. Based on this traceability matrices can be extracted, such as:

- SRS to URS
- SRS to Risks
- Risks to Mitigation
- SRS to Test

In addition our URS can also be traced to customer requirements.

SOUP Qualification

SOUP items (software of unknown provenance) have to be validated before use in the software system. To define an appropriate level of validation for each SOUP item the hazardous situations identified in the system risk analysis have to be traced to the software items. This is done in the Software Architecture Documentation. The level and scope of the validation of a SOUP item is then defined by its use in the identified software items.

For each SOUP item and associated hazard, the causes of the contribution are identified. Typical causes of SOUP items contributing to hazards include:

- Incorrect results from the SOUP item
- Insufficient performance of the SOUP item
- Incorrect error management of the SOUP item

To identify causes, any known anomalies documented in the release notes provided by the SOUP vendor are analyzed.

In the SOUP Qualification all SOUP items used by the ImFusion components of the product software system are listed. For each SOUP item, a brief description is given and the functional requirements as well as the safety classification in the context of the product software is described.

For each developed software system, a safety class is assigned. When a new development is initiated, the software system is systematically considered as a class C until the initial risk analysis has been performed. Then, based on the risk analysis, the safety class is adjusted according to the requirements of IEC 62304:

Class	*Description*
A	No injury or damage to health is possible as a result of a software malfunction
B	Non-serious injury is possible as a result of a software malfunction
C	Death or serious injury is possible as a result of a software malfunction

The safety classification has a direct impact on the level of scrutiny to be associated to the development process. In case of class A no further risk mitigation is required. For software safety class B and C, additional qualifications need to be applied. In all cases, the rationale that sustains the safety classification is documented in the SOUP Qualification and is reviewed upon any significant modification to the risk analysis.

Implementation Review Report

The Implementation Phase is closed with the Implementation Review Report. It contains a review checklist with which the completeness of the documents of the Implementation Phase is checked. Additionally, it is checked whether the SRS are completely covered by test cases and whether these test cases are correctly implemented with respect to their URS or SRS, respectively. Furthermore, the SOUP qualification is reviewed for completeness and for whether all SOUPs have been qualified successfully. Finally, the successful passing of the unit and static tests is confirmed.

Verification Phase

The verification phase is a critical step that focuses on systematically testing and confirming that the software meets its predefined requirements, specifications, and regulatory standards. This phase ensures the reliability, safety, and quality of the software before it proceeds to the validation phase and eventual deployment. Key activities in the verification phase include test planning, testing (unit, integration and functional testing) and documentation review.

The Verification Phase consists of the following documents:

- Software Test Specification
- Test Report
- Verification Review Report

Software Test Specification and Test Report

The verification and validation process is executed in successive steps, where the corresponding development outputs at each step are tested. Such in-depth verification (white box testing), rather than just a functional testing on the compiled software system (black box testing), permits to identify coding errors or defects that may not have an immediate visible impact on software functionalities and/or to identify them at an earlier stage in the development process.

The second level is the software system verification. At this level, the software system is executed in a test environment that represents the final environment of use (i.e. on a desktop PC, on a tablet, on the electronic board, etc.). At this stage, each function of the system is tested in order to verify that each requirement and risk control measure has been implemented as specified.

In the Test Report the result of the tests is documented with execution date, name of the tester, source code revision and test revision.

Verification Review Report

The Verification Phase is closed with the Verification Review Report. It contains a review checklist with which the completeness of the documents of the Verification Phase is checked. Additionally, here the successful implementation and performance of the test cases in an representative environment is verified. Furthermore, the correct documentation of the test results is checked.

Validation Phase

The validation phase is a critical step that focuses on confirming that the software meets its intended clinical and operational needs in the real-world environment. This phase ensures that the product performs as expected and is safe for use. Key activities in the validation phase include real-world simulation, usability validation and performance verification. The validation phase is the last crucial step before the deployment of the SaMD in clinical and healthcare settings.

Medical Devices with Artificial Intelligence

Medical devices containing artificial intelligence (AI) must fulfil additional regulatory and safety requirements. These include the technical description of the trained models (model report) and a cybersecurity analysis not only aiming for patient security but for data security.

Model Report

The Model Report describes which step during a workflow is done by an AI model and how the training of this model was performed. For the training of AI models the used data set is divided into training, validation and test data. The report described the key in- and outputs, how the data is divided into the training subsets and how the data was assessed, labelled and reviewed. Additionally, the training parameters are described in detail, containing tuning and hyperparameters as well as pre-processing, e.g. conversions, transformations, aggregations or normalization, and post-processing activities. Furthermore, the model reports states any limitations, software, hardware or performance requirements and interactions or dependencies. Finally, it is described how the model is tested and which metrics are used to measure accuracy and robustness.

Conclusion

This white paper has outlined the key deliverables of our technical documentation process in accordance with ISO 13485 standards.

The technical documentation serves as a valuable resource, offering essential information for product understanding, regulatory adherence, and seamless integration.

We from ImFusion stand as your reliable partner for medical software solutions. Our expertise spans from development in accordance with IEC 62304 standards to providing comprehensive technical documentation for the software we develop. With our dedicated team, we are committed to delivering innovative and high-quality solutions that not only meet regulatory requirements but also align with your specific needs. Contact us to learn more about how we can assist you in achieving your medical software goals.

Disclaimer

This white paper is provided for informational purposes only and should not be construed as legal advice. The information contained herein is subject to change without notice and is not guaranteed to be complete, accurate, or up-to-date. The opinions expressed in this white paper are those of the authors and do not necessarily reflect the views of regulatory authorities. ImFusion is not responsible for any errors or omissions in this white paper, nor for any direct, indirect, incidental, consequential, or other damages arising from or in connection with the use of, or reliance upon, this white paper or the information contained herein. Any reference to a specific product, service, or company in this white paper does not constitute an endorsement or recommendation by ImFusion. Users of this white paper are solely responsible for their use of the information contained herein and are encouraged to seek professional advice before taking any action based on the information provided.

Published March 2024

ImFusion GmbH

Agnes-Pockels-Bogen 1
80992 Munich, Germany

Phone: +49 (0)89 4524 6780
info@imfusion.com
www.imfusion.com