TECHNICAL REPORT

ISO/TR 80002-2

First edition
2017-06

Medical device software —

Part 2:
Validation of software for medical device quality systems

Logiciels de dispositifs médicaux —

Partie 2: Validation des logiciels pour les systèmes de qualité des dispositifs médicaux

Reference number
ISO/TR 80002-2:2017(E)

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO should not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 210, *Quality management and corresponding general aspects for medical devices*, in collaboration with Technical Committee IEC/TC 62, *Electrical equipment in medical practice*, Subcommittee SC 62A, *Common aspects of electrical equipment used in medical practice*, in accordance with ISO/IEC mode of cooperation 4.

A list of all parts in the ISO 80002 series can be found on the ISO website.

Introduction

This document has been developed to assist readers in determining appropriate activities for the validation of process software used in medical device quality systems using a risk-based approach that applies critical thinking.

This includes software used in the quality management system, software used in production and service provision, and software used for the monitoring and measurement of requirements, as required by ISO 13485:2016: 4.1.6, 7.5.6 and 7.6.

This document is the result of an effort to bring together experience from medical device industry personnel who deal with performing this type of software validation and who are tasked with establishing auditable documentation. The document has been developed with certain questions and problems in mind that we all go through when faced with validating process software used in medical device quality systems such as the following: What has to be done? How much is enough? How is risk analysis involved? After much discussion, it has been concluded that in every case, a set of activities (i.e. the tools from a toolbox) was identified to provide a level of confidence in the ability of the software to perform according to its intended use. However, the list of activities varied depending on factors including, among others, the complexity of the software, the risk of harm involved and the pedigree (e.g. quality, stability) of vendor-supplied software.

The intention of this document is to help stakeholders, including manufacturers, auditors and regulators, to understand and apply the requirement for validation of software included in ISO 13485:2016, 4.1.6, 7.5.6 and 7.6.

Medical device software —

Part 2: Validation of software for medical device quality systems

1 Scope

This document applies to any software used in device design, testing, component acceptance, manufacturing, labelling, packaging, distribution and complaint handling or to automate any other aspect of a medical device quality system as described in ISO 13485.

This document applies to

— software used in the quality management system,

— software used in production and service provision, and

— software used for the monitoring and measurement of requirements.

It does not apply to

— software used as a component, part or accessory of a medical device, or

— software that is itself a medical device.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO 9000 and ISO 13485 apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— IEC Electropedia: available at http://www.electropedia.org/

— ISO Online browsing platform: available at http://www.iso.org/obp

4 Software validation discussion

4.1 Definition

The term “software validation” has been interpreted both broadly and narrowly, from just testing to extensive activities including testing. This document uses the term software validation to denote all of the activities that establish a level of confidence that the software is appropriate for its intended use and that it is trustworthy and reliable. The chosen activities, whatever they might be, should ensure that the software meets its requirements and intended purpose.

4.2 Confidence-building activities: Tools in the toolbox

The tools in the toolbox (see Table A.1 to Table A.5) include activities completed during the life cycle of software that reduce risk and build confidence.

4.3 Critical thinking

This document promotes the use of critical thinking to determine which activities should be performed to adequately validate specific software. Critical thinking is a process of analysing and evaluating various aspects of software, as well as the environment in which it will be used, to identify the most meaningful set of confidence-building activities to be applied during validation. Critical thinking avoids an approach that applies a one-size-fits-all validation solution without thoroughly evaluating the solution to determine if it indeed results in the desired outcome. Critical thinking recognizes that validation solutions can vary greatly from software to software and also allows for different validation solutions to be applied to the same software in a similar situation. Critical thinking challenges proposed validation solutions, to ensure that they meet the intent of the quality management system requirements, and considers all key stakeholders and their needs. Critical thinking is also used to re-evaluate the validation solution when characteristics of the software change, when the software's intended use changes or when new information becomes available.

Critical thinking results in a validation solution that establishes compliance for a manufacturer, ensures that the software is safe for use, results in documented evidence that is deemed appropriate and adequate by reviewers, and results in a scenario in which individuals performing the validation work feels that the effort adds value and represents the most efficient way to reach the desired results.

Annex C presents example studies demonstrating how critical thinking can be applied to software validation of software used in medical device quality systems in a variety of situations, including different complexities, pedigrees and risk levels.

5 Software validation and critical thinking

5.1 Overview

Throughout the life cycle of software for medical device quality systems, appropriate controls need to be in place to ensure that the software performs as intended. Incorporation of critical thinking and application of selected confidence-building activities result in establishing and maintaining a validated state of the software. Figure 1 depicts a conceptual view of typical activities and controls that are part of the life cycle from the moment the decision is made to automate a process until the software is retired or is no longer used for medical device quality systems. Although Figure 1 depicts a sequential model, in reality, the process is of an iterative nature as elements are defined, risks are identified and critical thinking is applied.

When developing software for use in the medical device quality system, a fundamental confidence-building activity to be selected from the toolbox is the choice of software development life-cycle model. The model chosen should include critical thinking activities that enable the selection of other appropriate tools during various life-cycle activities. The results of the analyses and evaluations used drive the selection of the most meaningful set of confidence-building activities to ensure that the software performs as intended. This document does not mean to imply or prescribe the use of any particular software development model. For simplicity, however, the remainder of this document explains the concepts of critical thinking within the context of a waterfall development model using generic names for the phases. Other software development models (e.g. iterative, spiral) can certainly be used as long as critical thinking and the application of appropriate tools are incorporated into the model.

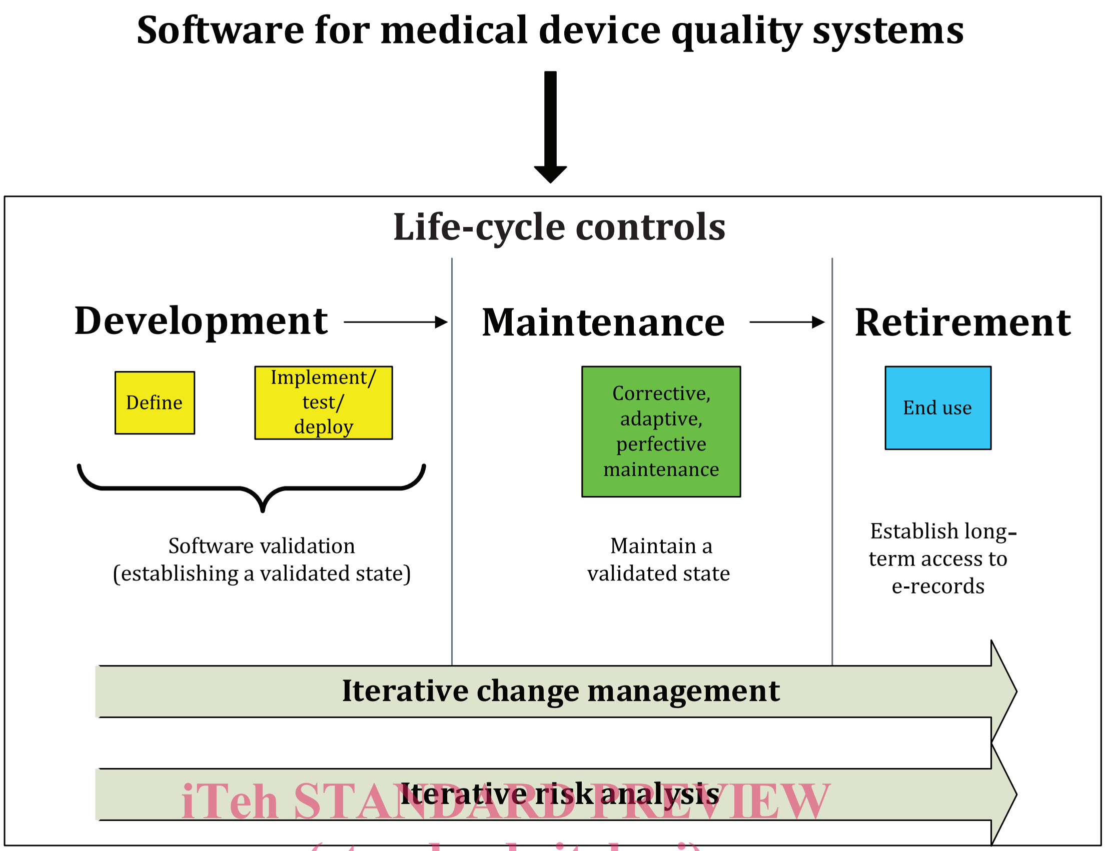

Software for medical device quality systems
Life-cycle controls
Development
Maintenance
Retirement
Define
Implement/ test/ deploy
Corrective, adaptive, perfective maintenance
End use
Software validation (establishing a validated state)
Maintain a validated state
Establish long-term access to e-records
Iterative change management
Iterative risk analysis

Figure 1 — Life-cycle controls

When considering using software in a process, one should identify whether the proposed software is used as part of a medical device quality system process through an investigation of its intended use. If so, then the software should be validated for its intended use. Although this document describes an approach to validating software for medical device quality systems, the same approach is also good practice for software to evaluate whether it fulfils defined requirements. The most critical part of software validation is developing/purchasing the right software tool to be able to support processes as intended by the manufacturer. This implies that requirements should be determined accurately to evaluate whether the developed/purchased software is suitable to fulfil the requirements of the intended use. Technical requirements suitable for verification, as well as process requirements suitable for validation, are equally important. When considering using software in a process, the software can interact or can have interfaces with other software.

During the development phase of the life cycle, risk management and validation planning tasks are performed to gather information and drive decisions in the following four areas:

— level of effort applied and scrutiny of documentation and deliverables;

— extent of content in the documentation and deliverables;

— selection of tools from the toolbox and methods for applying the tools;

— level of effort in applying the tools.

The primary drivers for decisions in the four areas are process risk and software risk. However, other drivers can influence decisions, including the complexity of the software and process, the type of software and the software pedigree.

The validation planning process consists of two distinct elements. The first validation planning element involves determining the level of rigor in the documentation and the scrutiny to be applied to the review of the resulting deliverables. The decisions in this element are primarily driven by the results of the process risk analysis. The second validation planning element drives the selection of tools from the toolbox to implement, test and deploy the software. The choice of tools is driven primarily by the software risk analysis. Such planning steps result from different types of risk analyses and are depicted as separate activities in this document. However, many times the steps are combined into one activity, which includes the different aspects of risk analysis and the resultant choices for proceeding with validation.

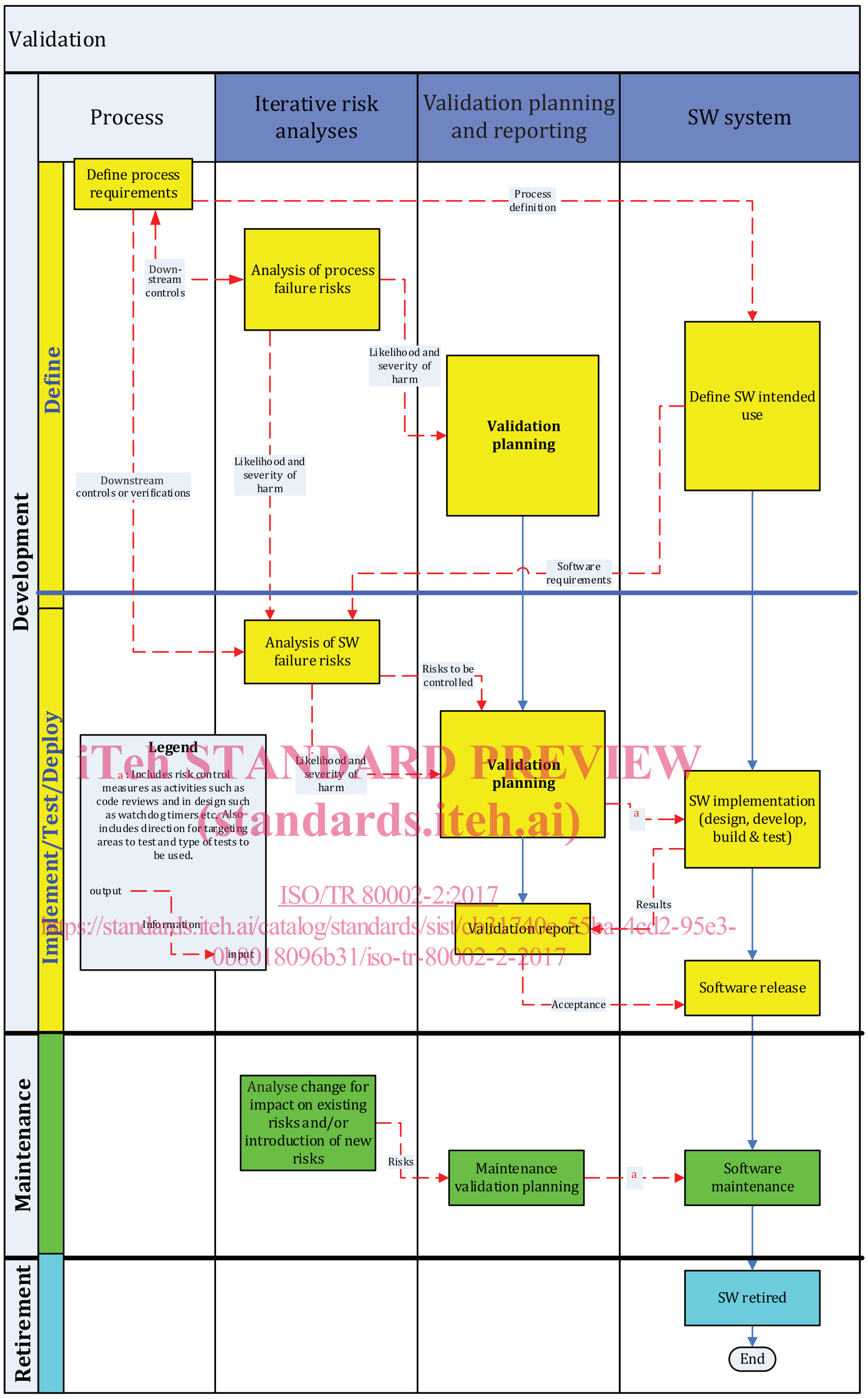
During the development phase of the life cycle, risk management and validation planning tasks are used to define the appropriate level of effort to be applied to the software and to determine what confidence-building tools to apply. This type of approach results in the completion of appropriate value-added activities and verification tasks, which are the basis for establishing a validated state. Once these activities and tasks are executed, the tools and their associated results are cited in a validation report as support for the conclusion that the software is validated.

Once deployed, the software moves into the maintenance phase of the software life cycle. During this period, the software is monitored, enhanced and updated as dictated by the business needs or regulatory requirement changes. Change control activities use the same concepts as the initial approach that was applied during the development phase of the life cycle. Changes, however, are now assessed as to their effect on the intended use, on the risk of failure, on the risk control measures that were applied during the initial development and on any functionality of the software itself.

The retirement phase is the act of removing software from use either by removal of the process or by replacement of the software being used for the process.

The activities shown in Figure 1 reflect the primary software life-cycle control activities. Other work streams include project management, process development, vendor management (if applicable), and possibly others, depending on the software being implemented.

Figure 2 depicts software life-cycle control activities and critical thinking within the context of other work stream activities. The critical thinking activities appear in the iterative risk analysis and validation work streams. It is important to have clear and formal definitions of these work streams within the organization's business model to ensure that a program properly manages the software from both business and regulatory perspectives.


Validation
Process
Iterative risk analyses
Validation planning and reporting
SW system
Development
Define
Implement/Test/Deploy
Maintenance
Retirement
Define process requirements
Process definition
Down-stream controls
Analysis of process failure risks
Likelihood and severity of harm
Validation planning
Define SW intended use
Downstream controls or verifications
Likelihood and severity of harm
Software requirements
Analysis of SW failure risks
Risks to be controlled
Validation planning
Likelihood and severity of harm
a
SW implementation (design, develop, build & test)
Results
Validation report
Acceptance
Software release
Legend
a : Includes risk control measures as activities such as code reviews and in design such as watchdog timers etc. Also includes direction for targeting areas to test and type of tests to be used.
output
Information
input
Analyse change for impact on existing risks and/or introduction of new risks
Risks
Maintenance validation planning
a
Software maintenance
SW retired
End

NOTE When the term "develop" or "development" is used, it is about the development of a validated state of the software.

Figure 2 — Life-cycle controls work stream

The various colours depicted in Figure 2 correspond to the life-cycle portion that is shown in the overall approach flow chart in Figure 1. The red dashed lines indicate information that is outputted from one activity and that provides input to or helps drive decisions in another activity. The diagram demonstrates how the ordering of the activities is driven by the need to have input information before completing the activities that require the input. It is important to note that all the activities are completed irrespective of the size or complexity of the software being implemented. However, for larger or more complex software, such activities will most likely be discrete; for smaller or simpler software, many of those activities will be combined or completed simultaneously.

In summary, the critical thinking approach described a systematic method for identifying and including appropriate confidence-building activities or tools in various work streams to support the conclusions that the software is validated on release and that the validated state will be maintained until the software is retired.

The following subclauses provide additional details for each of the blocks found in the life-cycle controls depicted in Figure 1. The subclauses use the work stream depiction of iterative risk analyses, validation and software activities shown in Figure 2 to provide perspective on the various decision points and decision drivers that incorporate critical thinking.

5.2 Determine if the software is in scope

5.2.1 Document a high-level definition of the process and use of the software

The first step in determining whether the software is considered to be used for medical device quality systems is to document a high-level definition of the process and use of the software. This activity might seem of small value when it is readily known that the software is in scope and one is already embarking on defining the full intended use of the software. However, for situations in which such assumptions are less clear, documenting the process and use enables the clear determination as to whether the software is in scope. In addition, for identified out-of-scope software, such an activity can result in a rationale as to why the software is out of scope.

5.2.2 Regulatory use assessment

A regulatory use assessment can be used to determine whether the software is a “software for medical device quality system” and therefore falls within the scope of this document. Start by identifying the specific regulatory requirements that apply to the processes that use the software and the data records that are managed by the software. A series of questions can be used to help fully understand the role that the software plays in support of these regulations. The following types of questions should be considered.

a) Could the failure or latent flaws of the software affect the safety or quality of medical devices?

b) Does the software automate or execute an activity required by regulatory requirements (in particular, the requirements for medical device quality management systems)? Examples may include capturing electronic signatures and/or records, maintaining product traceability, performing and capturing test results, maintaining data logs such as CAPA, non-conformances, complaints, calibrations, etc.

A “yes” answer to any of the questions identifies software that is required to be validated and is within scope of this document.

At times it can be difficult to determine whether a process and corresponding software are part of the quality system. Some tools can have many degrees of separation from the actual medical device. Each organization should, therefore, carefully consider the circumstances surrounding such borderline software and should completely understand the impact of the failure of the software on the processes and, ultimately, on the safety and efficacy of any manufactured medical devices. When the answer is not certain, the best approach is to consider the software as in scope and to apply the approach defined in this document.

5.2.3 Processes and software extraneous to medical device regulatory requirements

When processes or software contain functionality that falls outside of medical device regulatory requirements, an analysis should be performed to determine which parts of the software are considered to be in scope and which parts are not in scope. Such decisions should be rationalized on the basis of the degree of integration between various components, modules and data structures of the software and in accordance with the compliance needs of the organization. This rationalization is especially important in the case of software used in support of the quality system, such as large, complex enterprise resource planning (ERP) software. ERP software can include functionality for non-medical device-regulated processes such as accounting and finance. Although such functionality can be crucial for business operations and have to meet certain government requirements (e.g. those of the Sarbanes-Oxley Act).

5.3 Development phase

5.3.1 Validation planning

The first part of the validation planning activity captured when critical thinking is applied, uses input from the process risk analysis (see Annex B) to establish the basis for the level of effort that should be applied to the documentation and to drive the choices of tools from the Define section of the toolbox (see Table A.1 to Table A.5). The second part uses input from the software risk analysis to drive the choices of the implement, test and deploy tools from the toolbox. Once executed, the activities and the validated state of the software are established, and evidence of the validation is documented in the validation report.

Many development life-cycle models can be applied during the development phase. None is advocated or recommended by this document; however, application of a controlled methodology is expected. Such a controlled methodology would be based on the concept of defining requirements (including intended use), before implementation, testing and deployment, which are fundamental to establishing the validation of the software for its intended use.

5.3.2 Define

5.3.2.1 Define block requirement

The activities completed within the define block include the definition of the process, the definition of the software intended use within that process and the planning for the level of validation effort based on the inherent risks identified within the process. Figure 3 depicts this portion of the development phase within the selected waterfall model example.

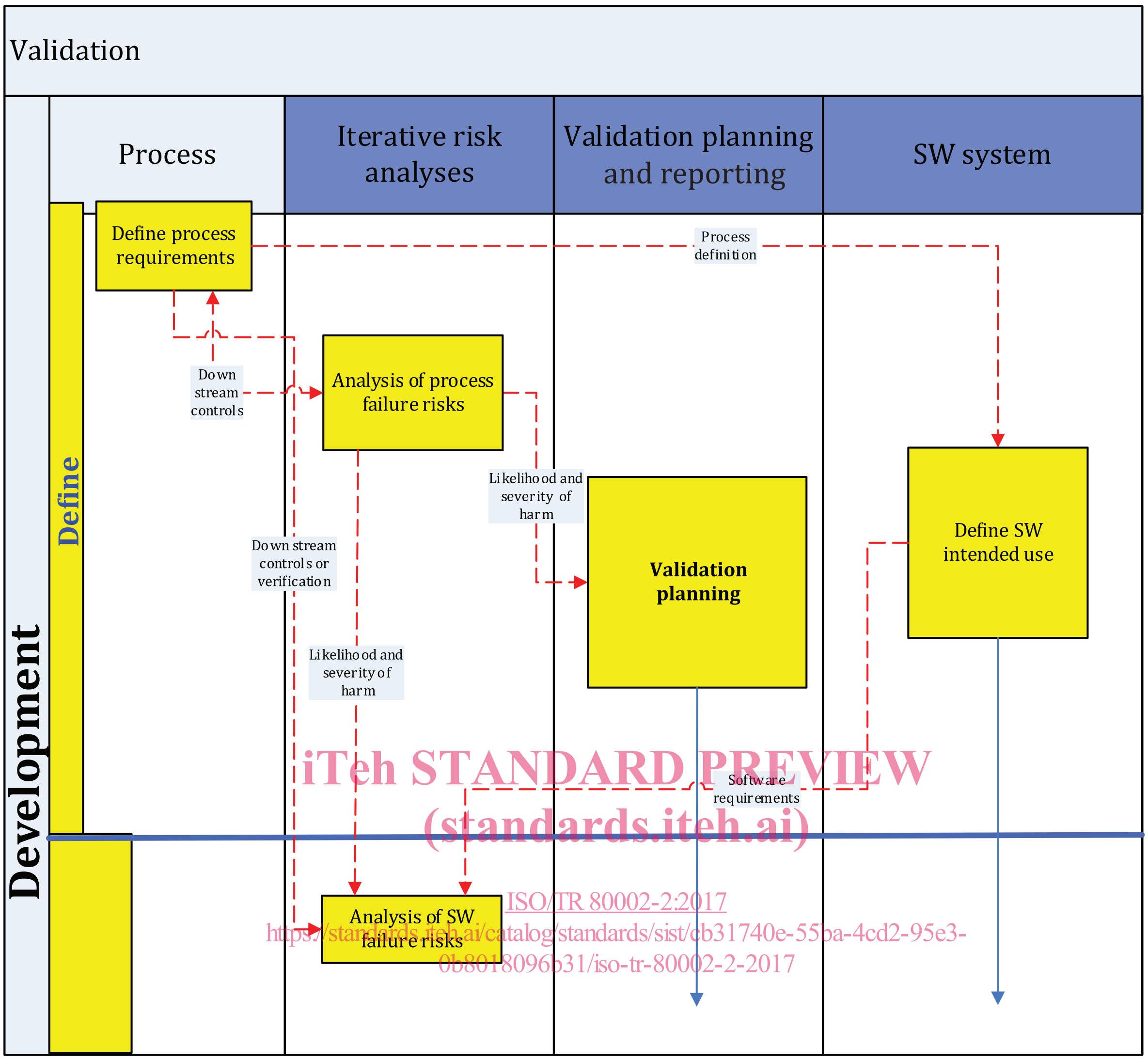

Validation
Process
Iterative risk analyses
Validation planning and reporting
SW system
Development
Define
Define process requirements
Process definition
Down stream controls
Analysis of process failure risks
Likelihood and severity of harm
Validation planning
Define SW intended use
Down stream controls or verification
Likelihood and severity of harm
Software requirements
Analysis of SW failure risks

Figure 3 — Life-cycle phase: Define block work streams

5.3.2.2 Process requirements

The first step in the application of life-cycle controls is to define the purpose and function of the entire process, particularly the portions intended to be controlled by the software. This is best performed by involving the appropriate subject matter experts and including all aspects and activities associated regardless of whether all will be controlled by the software. Benefits are explained below:

— regulatory requirements can be clearly discerned;

— intended use of the particular software within the context of the process can be clearly discerned;

— process aspects and activities not controlled by the particular software can be clearly identified and addressed procedurally or by some other means;

— process activities upstream and downstream from the software are identified and can be considered when assessing the risks of the software failure and in devising risk controls for software failure.

The process definition activity establishes the foundation for decisions that are made later in the life cycle and is essential to targeting efforts on value-added, risk-based activities.

5.3.2.3 Analysis of process failure risk

The relationship of the software to the final safety and efficacy of the medical product will be considered during the risk analysis process. The following should also be considered.

— Risk of harm to humans: This includes direct harm to patients and users, and indirect harm when software controlling manufacture or quality of the device malfunctions, resulting in failure of the device, which causes harm.

— Regulatory risk: Risk of non-compliance with regulatory requirements to be considered if failure of the software can lead to loss of records (e.g. CAPA, complaint, device master record or device history file records) required by regulatory agencies or to deviations from quality system and manufacturing procedures.

— Environmental risk: Risk to the environment in which the software operates. Both the physical and the virtual.

Other types of risks can be incorporated into this model. However, the scope of this document and the tools discussed to reduce risk do not address them. This document focuses on the determination of the human safety risks, regulatory risks and environmental risks associated with software failure within the context of process failure.

The results of risk analysis should be clearly documented because such results are valuable decision drivers for selecting tools from the toolbox and for justifying the level of effort applied to the validation activities.

5.3.2.4 Validation planning

The extent of confirmation and objective evidence needed to ensure that the requirements of the software can be consistently fulfilled depends on the critical value of the software within the overall process. Therefore, the first validation planning activity regarding the level of effort applied and the scrutiny of the deliverable elements is based solely on input from the process failure risk analysis.

This validation planning activity results in a first iteration of validation planning documentation. The planning includes the selections for “level of effort” (i.e. the decisions) and the rationale for those choices (i.e. the decision drivers). The rationale should be based on the risk of harm posed by a failure of the process. The validation plan should provide objective evidence of the application of critical thinking to the validation planning process.

5.3.2.5 Software intended use

5.3.2.5.1 Elements of intended use

The intended use is meant to provide a complete picture of the software functionality and its purpose within the process. Specifically, it is meant to describe and explain how the software fits into the overall process that it is automating, what the software does, what one can expect of the software and how much one can rely on the software to design, produce and maintain safe medical devices. The intended use is a key tool used to understand what potential risks are associated with the use of the software.

The three main elements of intended use are:

— purpose and intent related to

  — the software’s use (e.g. who, what, when, why, where and how),

  — the regulatory use of the software, and

  — the boundaries of the software within the process or with other software and/or users;

— software use requirements. As the complexity and, generally, the risk increases, this element adds more detailed information regarding the use of the software (e.g. use cases, user requirements);