62304 2nd Edition Change Rationales

Purpose

This document provides comment submission guidance and rationales for design decisions, and clause modifications (additive, removal, revision) made by IEC/SC 62A MT 49– Medical device software process.

NOTE: It is critical to fully read and follow the complete instructions provided to properly comment on the CD fragments for the comments to be accepted.

Forward

This rationale document is intended to be utilized in partnership with (1) the IEC SC62A Design Specification for the second edition of IEC 62304 and (2) the committee draft of IEC SC 62A/MT49 62304. The design specification was developed and approved by National Committees and IEC SC62A to establish the scope of the revision of IEC 62304. The experts appointed to IEC SC62A/MT49 have developed the current Committee Draft within the bounds of this approved design specification. This rationale document was developed to accompany the committee draft in order to provide the national committees reviewing the committee draft additional context derived from MT49 discussions to address and resolve the requirements within the approved design specification. National experts are encouraged to utilize both the approved design specification and rationale document when considering their comments and feedback which would be submitted back to MT49 on the proposed committee draft. Please keep in mind this committee draft is not to be considered technically complete but has been provided to assist MT49 with refining the approach to develop a mature committee draft for vote (CDV). The editing team also experienced challenges with the OSD tool in updating the linked connections to specific referenced clauses/subclauses within other standards found in Annex C of this document. MT49 will work to resolve this before the CDV stage.

We welcome comments will be collected through the Online Standards Development (OSD) platform where possible. Commenters are requested to submit only high-level technical comments on the documents. Editorial comments will not be considered at this time. There are glitches in the OSD platform, e.g., one that prevents correct capitalization of defined terms when used in the beginning of a sentence, loss of pluralization, and change of alternate terms to the full term (e.g., MEE to medical electrical equipment). In addition, the content of the fragments is in a very early stage of development and many changes are likely to be implemented prior to a CDV ballot of each fragment.

Each comment will appear within the OSD platform when properly formatted. It is essential that each comment is correctly identified by subclause number (e.g., 5.3.2.2.2.). Line number and list items (e.g., a) i)) should be listed in the text of the comment itself to allow identification of the comment's exact location in OSD. Comments that are not properly referenced in the comment template are likely to be ignored as the references are difficult to trace to the proposed text.

EXAMPLE:

- Green: the information in these cells is accepted (comment 1 is correctly formatted so that all of the relevant identifying information will be accepted into OSD.)

- Red: the information in these cells is not accepted by OSD (comment 2 is invalid and therefore not accepted by OSD because “a) i)” was added after subclause 4.1.1) **If any non-numeric text is included in the Clause/Subclause column, the comment will not be accepted**.)
- Orange: the information in these cells is not accepted (in comment 3, line number and item number will not be imported into OSD. Comments such as this one that are not properly referenced in the comment template are likely to be ignored as the references are difficult to trace to the proposed text.)

MB/ NC[1]	Line number (e.g. 17)	Clause/ Subclause (e.g. 3.1)	Paragraph/ Figure/ Table/ (e.g. Table 1)	Type of comment[2]	Comments	Proposed change	Observations of the secretariat
IT		4.1.1		Te	4.1.1 a) i) Line number: 50 Change the value of the temperature range to align with most current research	“Temperatures between -40 and 40 degrees”	
IT		4.1.1 a) i)		Te	Change the value of the temperature range to align with most current research	“Temperatures between -40 and 40 degrees”	
IT	50	4.1.1	a) i)	Te	Change the value of the temperature range to align with most current research	“Temperatures between -40 and 40 degrees”	

The intention of SC 62A is to allow experts and National Committees to gain experience using OSD and to provide visibility of the progress on the project to date. As the uncompleted sections are early in their development, full rationale and reference annexes are not fully prepared; there are informative box notes and notes with references to annexes that may not yet be complete.

Rationales

It is strongly recommended to utilize the content that follows with (1) the approved IEC SC62A Design Specification for the second edition of IEC 62304 and (2) the committee draft of IEC SC 62A/MT49 62304, when considering comments to submit.

4.1 Not used [ED 2.0]; *Quality management system [ED 1.1]*

Many of the proposals for the second edition of IEC 62304 recommended removing the quality system aspects requirements that are applicable to the overall development of a product (or the system). Since IEC 62304 is not a product standard, the requirement for a quality management system is left to the product (or system) level to define.

4.2 Risk management

The requirement for specific risk management process (or standard) was removed since risk management is done at the product (or system) level. Since IEC 62304 is not a product standard, the requirement for a specific risk management process is left to the product (or system) level to define.

4.3 Software process rigor [ED 2.0]; *Software safety classification [ED 1.1]*

Replacement of the three software safety classes with two software process rigor levels

In general, the 2nd edition aims at a simplification of the software classification to drive the software process rigor. The most obvious simplification is to have only two levels instead of three, where level I essentially replaces software safety class A and level II replaces software safety class B and C. A differentiation between class B and C is no longer intended, so that IEC 62304 aligns better with IEC 81001-5-1.

Abandoning the term “hazardous situation”

The term “hazardous situation” is a specific term of ISO 14971, and we wanted to allow for other product risk management processes. A hazardous situation can per definition potentially lead to harm (by exposing somebody or something to a hazard, which per definition is a potential source of harm). Therefore, asking whether a software failure can lead to harm is potentially easier to understand for a wider audience caused by the scope expansion into health software.

Using “harm” instead of “harm to patient, operator, or other people” (injury)

This aligns better with ISO 14971, which also addresses harm to property and harm to the environment.

Abandoning risk acceptance in the determination of process rigor

There has been much confusion regarding the use of ISO 14971's risk acceptance in the context of software safety classification. The confusion is considered even more significant with the scope extension. Hence, we have removed “unacceptable RISK” from the classification and replaced it with clearer guidance; see the next topic of **Meaning of “very unlikely” and "negligible harm“** for more information.

Meaning of “very unlikely” and "negligible harm“

As a replacement of “unacceptable RISK, we were looking for criteria to exclude risks with very low probability or very low severity. For the very low probability we used “very unlikely to occur”, which does not need further definition (some other standards propose “improbable” as the lowest probability level and explain it as “very unlikely”). The term “negligible” conforms with other standards where it is proposed as the lowest severity level. For example, ISO 24971 Table 4 does that and describes it as “Results in inconvenience or temporary discomfort”, which we reused in our definition.

Accepting risks with very low probability or very low severity only is a little stricter than usual risk acceptance schemes that usually also accept risks with low (but not very low) probability and low (but not very low) severity. This is intentional, aiming at a simple scheme.

Probability of software failures

IEC 62304 ed. 1.1 stated "probability of software failure is assumed to be 1". This statement has often been questioned because software isn’t always failing. However, to determine the software process rigor level, we must assume that software defects exist and can, at some point, manifest as software failures. So, instead of focusing on whether the probability of software failures is 1, we suggest working with worst-case scenarios. This change is not intended to ease the rigor compared to ed. 1.1. However, it is believed to simplify the language and make it easier for readers unfamiliar with ISO 14971 concepts.

Consideration of external risk control measures

It is acknowledged that the IEC 62304 ed2 design specification suggests not including external risk control measures in determining the software process rigor level.

However, the project team suggests keeping the possibility of leveraging external risk control measures as this is believed to be good engineering practice. Furthermore, we prefer keeping track of such risk control measures instead of risking that manufacturers might react by declaring risk control measures as part of the initial design.

To minimize the risk of external risk control measures being misused, we have added a requirement to verify the effectiveness of risk control measures used to lower the process rigor level.

4.4 Not used [ED 2.0]; *Legacy Software [ED 1.1]*

The previous requirements for *legacy software* showing conformity to IEC 62304 have been moved to an informative annex as a recommended method for *legacy software* conformity. This was done in response to several comments and concerns received during the previous attempt of an IEC 62304 second edition since claiming conformity to updated standards is a regulatory topic and therefore out of scope of this effort.

5 Software development process

5.1 Software Development Planning

5.1.1 Software development plan

Updated to amend Safety Classifications A, B and C to Software Rigor Levels I and II, in line with agreed proposals from national committees.

Addition of note 6 for clarification that the Software Development Plan (SDP) is applicable to all rigor levels though allows for justification where sections are not applicable.

5.1.2 Keep software development plan updated

The text remains the same with the exception of change from Safety Classifications A, B and C to Software Rigor Levels I and II, in line with agreed proposals from national committees.

5.1.3 Software development plan reference to system design and development

The text remains the same with the exception of change from Safety Classifications A, B and C to Software Rigor Levels I and II, in line with agreed proposals from national committees.

5.1.4 Software development standards, methods and tools planning

Removal of “of class C”, where this clause applies Software Rigor Level I and II from a previously stated Class C.

A note is added to clarify the intention of the word “*Method*” to mean “*any technique to support software development*”. These *methods* and *tools* are typically validated under Quality Management System requirements, such as Computer Software Validation or Computer Software Assurance programs within medical devices. The addition of these requirements are therefore intended to be for clarification purposes.

Section 5.1.4 is extended to clarify the requirement under 5.1.4. (b), where the manufacturer is required to specify which *method* is used in the following activities (all verification within software development)

- Requirements verification
- Architecture verification
- Design verification
- Unit verification
- Integration testing
- Software system verification

Section 5.1.4 c) is added to clarify that the manufacturer shall document each method according to the requirements of the rigor level

Software Rigor Level I and II is applied to this clause.

5.1.5 Software integration and integration testing planning – Change from Class B, C to Level I and II, in line with agreed proposals from national committees.

5.1.6 Software verification planning – Change from Class A, B, C to Level I and II, in line with the Design Specification

5.1.7 Software risk management planning - Change from Class A, B, C to Level I and II, in line with the Design Specification

5.1.8 Documentation planning – minor formatting change of a, b, _, d to a, b, c. and change from Class A, B, C to Level I and II, in line with the Design Specification

5.1.9. Software configuration management planning - Change from Class A, B, C to Level I and II, in line with agreed proposals from national committees.

5.1.10 Supporting items to be controlled – amended to clarify the intent of ED1 with ED2 being expanded to bullet out "Tools, Development environment, Settings, Data and Simulators" from that stated in ED1 as "tools items or settings". This is required due to the updates in technology and clarify that the Development environment, Data and Simulators that can impact health software should be also controlled. Addition of Note 3 clarifies that validation of non-product software is addressed by the quality system at product level (as mentioned above, typically through computers system validation/computer system assurance programs). Also, change from Class B, C to Level I and II, in line with agreed proposals from national committees.

5.1.11 Software configuration item control before verification - Change from Class B, C to Level I and II, in line with the Design Specification.

5.1.12 Identification and avoidance of common software defects – removed and now indicated as "Not Used".

5.1.13 Software release planning – new clause 5.1.13 to clarity the requirement for release planning to be done early in the development lifecycle. The new clause indicates that the "manufacturer shall define or reference the procedure for software release, according to the purpose of the release, in the software development plan. This requirement is assigned as having software rigor level I and II.

5.1.14 Software maintenance planning – new clause 5.1.14 to clarify the requirement for software maintenance planning to be initiated early in the software development lifecycle and states that "the manufacturer shall include or reference a software maintenance plan (see clause 6.1) in the Software Development Plan. This requirement is assigned as having software rigor level I and II. The need for improvement maintenance to be initiated early in the SDLC is critical to reducing defects in the market from late or poorly planned maintenance post-deployment. This requirement is assigned as having software rigor level I and II.

5.1.15 **Artificial Intelligence (AI) planning** – new clause 5.1.15 for AI devices that include AI (basic decision trees, logic gates AND/OR), Machine Learning (ML), Deep Learning (DL), which is a subset of ML though warrants mentioning, Reinforcement Learning (RL) or Large Language Models (LLM)s with the intention of encompassing Generative or General AI models. It is known that Generative Models are in development (e.g., transcribing doctor notes or prescriptions) and therefore, all AI algorithms are included in the scope of this requirement.

5.1.15.1 **AI process planning** specifies that where an AI algorithm of any type (as listed in 5.1.15) or where AI data is used in health software, AI process planning is required, which is broken down into an AI Development Lifecycle (AIDL). Refer to introduction for supporting information on this. The phases identified for AIDL are: AI-enabled Device (AIeD) Planning and Design, Data Engineering and Management, Model Building and Tuning, Verification and Validation, Model Deployment, Model Maintenance and Decommission. This requirement is assigned as having software rigor level I and II. The bullets within each of these phases identifies aspects that must be considered, planned and documented. Annex E is presented as an informative text which is intended to support the state of the art on many of these requirements, whilst accepting that standards are continuing to be developed ISO 24971-2, IEC 63450, IEC 63521 and Data Lifecycle for Healthcare, etc. The phases chosen are not a replica of other AI Development Lifecycles, but are representative of current state of the art, and are intended to be harmonised within other regulatory bodies to the extent possible without leaving out a critical process or activity to be considered. The manufacturer is required to describe the tasks that will be performed in line with the activities presented. It is accepted that the AIDL shall be integrated within the SDLC and the existing SDLC requirements shall remain as predicate engineering practices, therefore, the AIDL activities are superimposed on top of the existing SDLC requirements, unless adequately justified. For example, in the event that there is a medical device under development, the expectation is that the Software Development Plan and subsequent requirements will hold and be delivered, with the addition of an AI development plan for the AI relevant aspects of the project. Whilst the AI requirements are specified in the planning section of the standard, the intention is that the rest of the standard applies where appropriate and the text is broad enough in most instances to cover the AI requirements also. Also, it is noted that AI Continuous Monitoring is schematically represented as a complete process under Fig. 4 of the introduction, demonstrating (as noted in Annex E) that similar to the maintenance plan, the monitoring planning shall be initiated early in the development lifecycle to ensure that the monitoring tools/methods/metrics are defined and tested early and where possible coded into an AI system for automatic alerts/errors where degradation or drift is identified in the performance of the AI model. Manual checks are not sufficient to ensure appropriate monitoring in the post-market phase. It is important to recognize that both locked and continuous learning (adaptive) algorithms must be monitored where degradation and drift arise with locked algorithms also, though accepted not with the same rate of change. A risk-based approach will be accepted. This requirement is assigned as having software rigor level I and II.

5.1.15.2 **AI Documentation** calls for additional documentation required for AIeD. Note: the AI V&V Plan is in addition to the Software/System V&V Plan, unless justified. The AI V&V plan / report(s) can be combined where practicable. The Where the AI Verification / Validation Plan(s) or Report(s) are combined, the manufacturer will ensure that the purpose and content remains adequate and clear. This requirement is assigned as having software rigor level I and II.

5.1.15.3 **AI Performance Evaluation Planning** is a new clause that identifies the need for performance evaluation plan and report and shall take account of the 3 pillars of performance, clinical, technical/analytical and scientific. Notes are provided as guidance under each of these and reference is made to the standard under development IEC 63521 for Performance Evaluation for greater detail. This requirement is assigned as having software rigor level I and II.

5.1.16 **Communication plan** is a new clause that defines requirements for a communication strategy. This requirement is assigned as having software rigor level I and II.

5.2 **Software requirements analysis** – no change to heading.

5..2.1 **Define and document software requirement from system requirements**. The intent of the requirement remains the same whilst the wording has changed to reflect the broadening of the standard to include health software (which includes medical devices). The notes have been amended for clarity purposes. This requirement is assigned as having software rigor level I and II.

5.2.2 **Software requirements content.** The intent of the requirement remains the same whilst the wording has changed to reflect the broadening of the standard to include health software (including medical devices). Whilst the sub-clauses remain largely the same, with the exception of additional notes for clarity and removal of "regulatory requirements" which are driven from the product level and cause confusion when presented here. This requirement is assigned as having software rigor level I and II.

5.2.3 **Include risk control measures in software requirements**. The intent of the requirement remains the same whilst the wording has changed to reflect the broadening of the standard to include health software (including medical devices). This requirement is assigned as having software rigor level II.

5.2.4 **Re-evaluation product risk analysis**. The intent of the requirement remains the same whilst the wording has changed to reflect the broadening of the standard to include health software (including medical devices), including reflected in the title, which used to refer to "medical device risk analysis" in ED1. It now calls for re-evaluation of the product risk analysis when software requirements are established. This requirement is assigned as having software rigor level I and II.

5.2.5 **Update requirements**. The intent of the requirement remains the same whilst the wording has changed to reflect the broadening of the standard to include health software (including medical devices). This requirement is assigned as having software rigor level I and II.

5.2.6 **Verify software requirements.** No change with exception of the change from Class A, B, C to Level I and II, in line with agreed proposals from national committees.

5.3 **Software architecture design**. No change to heading.

5.3.1 **Transform software requirements in an architecture**. The intent of the requirement remains the same whilst the wording has changed to reflect the broadening of the standards scope to include health software (including medical devices). Additional notes have been added for clarity. Change from Class B, C to Level I and II, in line with agreed proposals from national committees.

5.3.2 **Develop and architecture for the interfaces of software item**(s). Minor change which includes clarifications in parentheses "(both software and hardware)" to clarify the requirement. Change from Class B, C to Level I and II, in line with agreed proposals from national committees.

5.3.3 **Specify function and performance requirements of SOUP item**. The intent of the requirement remains the same whilst the wording has changed to reflect the broadening of the standard to include health software (including medical devices). Change from Class B, C to Level I and II, in line with agreed proposals from national committees.

5.3.4 **Specify system hardware and software required by SOUP item**. Change from Class B, C to Level I and II, in line with agreed proposals from national committees.

5.3.5. **Identify segregation necessary for risk control**. Changed wording to remove the requirement for demonstrating effective segregation by replacing with "The manufacturer (3.13) shall document any segregation between software item (3.28) that is necessary for risk control (3.21), and document a rationale for the adequacy of the design and the technology chosen for segregation ". The note has also been updated for clarification purposes. Change from Class C to Level II for a least burdensome approach.

5.3.6 **Verify software architecture**. Change from Class B, C to Level I and II, in line with agreed proposals from national committees.

5.4 **Software design** – no change to title

5.4.1 **Subdivide software into software unit.** Updated to broaden the clause for health software and clarify the intent of ED1 typically documented within medical devices for example, by addition of specifying a design specification, commensurate with risk will document the software requirements that represent the architecture. Change from Class B, C to Level I and II, in line with agreed proposals from national committees.

5.4.2. **Develop design for each software unit**. New sub-clause "b) the manufacturer shall provide a rationale for selecting from the design methods documented in the SDP". Change from Class C to Level II for a least burdensome approach.

5.4.3 **Develop detailed design for interfaces.** Removal of the term "detailed" from the heading to reflect the position from industry and agreed by ballot that it is confusing to differentiate between design and detailed design and can be removed. The industry experts desired to document the design level necessary to reflect the architecture and design and do not require the term "detailed" as it can be subjective. The content of the requirement remains unchanged. Change from Class C to Level II for a least burdensome approach.

5.4.4 **Verify design**. Removal of the term "detailed" from the heading and body of the requirement as described under 5.4.3. The requirement remains unchanged. Change from Class C to Level II for a least burdensome approach.

5.5 **Software unit implementation**. No change to heading.

5.5.1 **Implement each software unit**. No change with exception of the change from Class A, B, C to Level I and II, in line with agreed proposals from national committees.

5.5.2 **Establish software unit verification process**. The requirement has been amended to remove "establish strategies, methods and procedures" for verifying software units and is replaced by "shall perform unit verification as documented in the SDP". The requirement to test procedures for adequacy remains and guidance on this is provided. The change takes a least burdensome approach and avoids confusion as to what is intended by "strategies, methods and procedures". Change from Class A, B, C to Level I and II, in line with agreed proposals from national committees.

5.5.3 **Software unit acceptance criteria**. Additional notes for clarification. No change to requirement with exception of the change from Class B, C to Level II, for least burdensome approach.

5.5.4 **Additional software unit acceptance criteria**. The requirement remains unchanged. Change from Class C to Level II for a least burdensome approach.

5.5.5 **Software unit verification.** The requirement remains unchanged. Change from Class B and C to Level II for a least burdensome approach.

5.5.6 **Static Code Analysis**. New clause added to ensure for Level II "the manufacturer shall perform Static Code Analysis as part of the unit verification). A note is provided indicating that this can be automated.

5.6 **Software integration and integration testing** – no change to heading.

5.6.1 **Integrate software unit.** The requirement remains unchanged. Change from Class B, C to Level I and II, in line with the Design Specification.

5.6.2 **Verify software integration**. The requirement remains unchanged. Change from Class B and C to Level II for a least burdensome approach.

5.6.3 **Software integration testing**. No change with exception of the change from Class A, B, C to Level I and II, in line with the Design Specification.

5.6.4 **Software integration testing content**. The requirement remains unchanged. Change from Class B and C to Level II for a least burdensome approach.

5.6.5 **Evaluate [Evaluation] software integration test procedures**. The requirement remains unchanged. Change from Class B and C to Level II for a least burdensome approach.

5.6.6 **Conduct regression tests**. No change with exception of the change from Class A, B, C to Level I and II, in line with the Design Specification.

5.6.7 **Integration test record contents**. Minor change without change to the intent. Sub-clause C) clarifies the requirement from "identify the tester" to "document the identity of the person(s) responsible for executing the test as well as recording… . Change from Class B, C to Level I and II, in line with the Design Specification.

5.6.8 **Use software problem resolution process**. The requirement remains unchanged. Change from Class B and C to Level II for a least burdensome approach.

5.7 **Software system testing –** no change to heading.

5.7.1 **Establish test for software requirements**. No change with exception of the change from Class A, B, C to Level I and II, in line with the Design Specification.

5.7.2 **Use software problem resolution process**. No change with exception of the change from Class A, B, C to Level I and II, in line with the Design Specification.

5.7.3 **Retest after changes**. No change with exception of the change from Class A, B, C to Level I and II, in line with the Design Specification.

5.7.4 **Evaluate software system testing**. The term "adequacy" has been replaced by the term "appropriateness" with guidance on the extent of adequacy available in annexes. No other change with exception of the change from Class A, B, C to Level I and II, in line with the Design Specification.

5.7.5 **Software system test record contents**. No change with exception introduction of Level I and II, in line with the Design Specification. A, B, C. previously omitted in error in ED1.

5.8 **Software release** – no change to heading.

5.8.1 **Ensure software verification is complete**. No change to requirement with exception of change from Class A, B, C to Level I and II, in line with the Design Specification.

5.8.2 **Document known residual anomaly** [anomalies]. No change to requirement with exception of change from Class A, B, C to Level I and II, in line with the Design Specification.

5.8.3 **Evaluate [Evaluation] known residual anomalies** [anomaly] No change to requirement with exception of change from Class B and C to Level II for a least burdensome approach.

5.8.4 **Document released version**. Minor change to terms of reference from medical device to "health software" to broaden scope of standard and requirement. No change to the intent of the requirement with exception of change from Class B and C to Level II for a least burdensome approach.

5.8.5 **Document how released software was created**. No change to the requirement with exception of change from Class B and C to Level II for a least burdensome approach.

5.8.6 **Ensure activities and tasks are complete**. Sub-clause a) remains as previous ED1 with addition of new sub-clause b) to clarify the original intent, where the manufacturer must ensure reproducibility, identification, integrity, comprehensive release, support of software deployment, rollback plan and maintenance/support of released software. This extends the original requirement taking account of new technologies requiring maintenance and support and to avoid deployment / maintenance failures. Change from Class B, C to Level I and II, in line with the Design Specification.

5.8.7 **Archive software**. Minor change to terms of reference for Health Software in line with the broadening of the scope of the standard. No change to the intent of the requirement with exception of change from Class A, B, C to Level I and II, in line with the Design Specification.

5.8.8 **Assure reliable delivery of released software**. Minor change to terms of reference for Health Software in line with the broadening of the scope of the standard. No change to the intent of the requirement with exception of change from Class A, B, C to Level I and II, in line with the Design Specification.

6 Software maintenance process

The new revision carefully considers the definitions for Software maintenance and makes clear that "software maintenance" does not include adding/supporting new features since this should be done using the development process.

The definition of 'software maintenance' in IEC 82304 has been revised in its source ISO/IEC 14764 and was the basis for revision of clause 6. ISO/IEC 14764 (JTC1 SC7) has been revised in 2022, with updates to definitions.

Clause 6.1 makes clear that other than maintenance activities need to follow the complete design steps of the revised standard.

The software development maintenance plan has appropriate clause 5 activities based on each type of maintenance (corrective, adaptive, perfective, preventive).

Clause 6.2 was modified for maintenance to remove quality system level requirements. However, there may be elements to hand off tasks between the system and the product level software implementation.

Since product-level processes are not specified in the revised IEC 62304, any communication tasks during maintenance, with the intent to inform parties external to the software development organization, the revised IEC 62304 will be agnostic regarding the targets of such communication: In case health software is a SaMD or SiMD, the 'user' role would be the medical professional, in case of health software being administrative software (see definition of Health Software), the 'user' is a clinical professional at large and in case of health software being a sub-system within a combined product (incorporating different software systems), the 'user' role will be held by the product owner of the manufacturer of that product.

As a result, revised Clause 6 cannot make any assumptions about what entity (external to the software team) would be the external party to inform about events or modifications regarding health software. Therefore, any dependencies from the software to some external organization are resolved via communication roles that have to be defined as a part of the >product< maintenance plan. In clause 6 therefore the target of software-related notifications is a role name – (e.g. 'user role') together with a requirement to define that user role as a part of the product maintenance plan. In

case such role (‘authority role’) describes entities related to a specific market, the software maintenance plan can also refer to the product maintenance plan, which then resolves the names of the respective authorities for all markets targeted by the product.

The software maintenance plan specified in sub-section 6.1 mentions these roles to be resolved in the product maintenance plan.

7 Software risk management process

While risk management is performed at the product (or system) level, there is risk management information that needs to be shared between the system and software developers. This clause focuses on information to be provided by the software developers to ensure complete documentation, risk assessment and evaluation of the health software in the product (system).

8 Software configuration management process

The Design Specification does not suggest changes other than stating that requirements are applicable for all “process rigor levels”, where that term had been all “safety classes” in the currently published version.

9 Problem resolution process

The Design Specification does not suggest changes other than considering some updates done in the draft of 2020 (IEC 62A/1349/CDV).

These changes are mainly stating that requirements are applicable for all “process rigor levels”, where that term had been all “safety classes” in the currently published version. There are two clarifications in the CDV for 9.8