IEC TR 80002-3

Edition 1.0 2014-06

INTERNATIONAL STANDARD

Medical device software –
Part 3: Process reference model of medical device software life cycle processes (IEC 62304)

IEC Central Office
3, rue de Varembé
CH-1211 Geneva 20
Switzerland

Tel.: +41 22 919 02 11
Fax: +41 22 919 03 00
info@iec.ch
www.iec.ch

About the IEC
The International Electrotechnical Commission (IEC) is the leading global organization that prepares and publishes International Standards for all electrical, electronic and related technologies.

About IEC publications
The technical content of IEC publications is kept under constant review by the IEC. Please make sure that you have the latest edition, a corrigenda or an amendment might have been published.

IEC Catalogue - webstore.iec.ch/catalogue
The stand-alone application for consulting the entire bibliographical information on IEC International Standards, Technical Specifications, Technical Reports and other documents. Available for PC, Mac OS, Android Tablets and iPad.

IEC publications search - www.iec.ch/searchpub
The advanced search enables to find IEC publications by a variety of criteria (reference number, text, technical committee,…). It also gives information on projects, replaced and withdrawn publications.

IEC Just Published - webstore.iec.ch/justpublished
Stay up to date on all new IEC publications. Just Published details all new publications released. Available online and also once a month by email.

Electropedia - www.electropedia.org
The world's leading online dictionary of electronic and electrical terms containing more than 30 000 terms and definitions in English and French, with equivalent terms in 14 additional languages. Also known as the International Electrotechnical Vocabulary (IEV) online.

IEC Glossary - std.iec.ch/glossary
More than 55 000 electrotechnical terminology entries in English and French extracted from the Terms and Definitions clause of IEC publications issued since 2002. Some entries have been collected from earlier publications of IEC TC 37, 77, 86 and CISPR.

IEC Customer Service Centre - webstore.iec.ch/csc
If you wish to give us your feedback on this publication or need further assistance, please contact the Customer Service Centre: csc@iec.ch.

IEC TR 80002-3

Edition 1.0 2014-06

INTERNATIONAL STANDARD

Medical device software –
Part 3: Process reference model of medical device software life cycle processes (IEC 62304)

INTERNATIONAL
ELECTROTECHNICAL
COMMISSION

PRICE CODE U

ICS 11.040.01

ISBN 978-2-8322-1616-3

Warning! Make sure that you obtained this publication from an authorized distributor.

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

____

MEDICAL DEVICE SOFTWARE –

Part 3: Process reference model of medical device software life cycle processes (IEC 62304)

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.

2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.

3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.

4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.

5) IEC itself does not provide any attestation of conformity. Independent certification bodies provide conformity assessment services and, in some areas, access to IEC marks of conformity. IEC is not responsible for any services carried out by independent certification bodies.

6) All users should ensure that they have the latest edition of this publication.

7) No liability shall attach to IEC or its directors, employees, servants or agents including individual experts and members of its technical committees and IEC National Committees for any personal injury, property damage or other damage of any nature whatsoever, whether direct or indirect, or for costs (including legal fees) and expenses arising out of the publication, use of, or reliance upon, this IEC Publication or any other IEC Publications.

8) Attention is drawn to the Normative references cited in this publication. Use of the referenced publications is indispensable for the correct application of this publication.

9) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

The main task of IEC technical committees is to prepare International Standards. However, a technical committee may propose the publication of a technical report when it has collected data of a different kind from that which is normally published as an International Standard, for example "state of the art".

IEC TR 80002-3, which is a technical report, has been prepared by a Joint Working Group of subcommittee 62A: Common aspects of electrical equipment used in medical practice, of IEC technical committee 62*:* Electrical equipment in medical practice, and ISO technical committee 210: Quality management and corresponding general aspects for medical devices. It is published as a double logo standard.

The text of this technical report is based on the following documents:

Enquiry draft	Report on voting
62A/918/DTR	62A/928/RVC

Full information on the voting for the approval of this technical report can be found in the report on voting indicated in the above table. In ISO, the technical report has been approved by 14 P members out of 16 having cast a vote.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 2 and in accordance with ISO/IEC 24774, *Systems and software engineering – Life cycle management – Guidelines for process description*.

A list of all parts of the IEC 80002 series, published under the general title *Medical device software,* can be found on the IEC website.

The committee has decided that the contents of this publication will remain unchanged until the stability date indicated on the IEC web site under "http://webstore.iec.ch" in the data related to the specific publication. At this date, the publication will be

- reconfirmed,
- withdrawn,
- replaced by a revised edition, or
- amended.

A bilingual version of this publication may be issued at a later date.

INTRODUCTION

0.1 Background

Software is often an integral part of medical device technology. Establishing the safety and effectiveness of a medical device containing software requires well designed software that fulfils its purpose without causing any unacceptable risks. Following an internationally approved set of software development practices provides one way of achieving this.

This technical report (TR) provides a framework of life cycle processes supporting the safe design and maintenance of medical device software called the process reference model (PRM). The process descriptions in this PRM are fully compliant with the requirements of ISO/IEC 24774:2010, *Systems and software engineering – Life cycle management – Guidelines for process description*.

This TR presents the PRM for medical device software development as a result of integrating requirements from IEC 62304:2006 and from the international standard of software life-cycle processes ISO/IEC 12207:2008.

This TR is aimed at medical device software developers who can use it for realizing the set of requirements they have to achieve to be compliant with IEC 62304:2006 in the scope of the safety class of the medical device software they are developing. Each process outcome with a corresponding safety class is a requirement in IEC 62304:2006. The process outcomes without a corresponding safety class are based only on ISO/IEC 12207:2008. These process outcomes provide additions that are beneficial when achieving the purpose of the process and could be regarded as a valuable contribution to safety-critical software development. The PRM may also be used to provide a common basis for different models and methods for process assessment, ensuring that the results of the assessments can be reported in a common context. Assessors who are concerned with examining medical device software processes can use the PRM as an agreed list of IEC 62304 process outcomes to inform audit check listing and reporting.

The process descriptions in the PRM incorporate a statement of the purpose of the process which describes at a high level the overall objectives of performing the process, together with the set of outcomes which demonstrate the successful achievement of the process purpose. These process outcomes are the software life cycle process requirements – the statements of the overall goal of performing the process. In any process description, the set of process outcomes are necessary and sufficient to achieve the purpose of the process.

A manufacturer of a medical device software system is required to assign a software safety class (A, B, or C) according to the possible effects on the patient, operator, or other people resulting from a hazard to which the software system contributes, described in greater detail in IEC 62304:2006. The software safety classes are assigned based on severity as follows:

- Class A: no injury or damage to health is possible;
- Class B: non-serious injury is possible;
- Class C: death or serious injury is possible.

0.2 Organization of this technical report

This TR is organized to follow the structure of IEC 62304. Annex A describes the development of the TR in greater detail. Annex B provides a mapping from IEC 62304 clauses together with their safety classes to the corresponding ISO/IEC 12207:2008 processes. The life cycle processes of the PRM for medical device software development are described in terms of process name, process purpose and the corresponding process outcomes. The outcomes marked with an "[ISO/IEC 12207]" at the end of the outcome statement are derived from ISO/IEC 12207:2008, with no directly corresponding requirement in IEC 62304. Users of this PRM who wish to examine only the IEC 62304 requirements can elect to disregard the outcomes that are based only on ISO/IEC 12207:2008.

MEDICAL DEVICE SOFTWARE –

Part 3: Process reference model of medical device software life cycle processes (IEC 62304)

1 Scope

This part of IEC 80002, which is a technical report (TR), provides the description of software life cycle processes for medical devices. The medical device software life cycle processes are derived from IEC 62304:2006, with corresponding safety classes. They have been aligned with the software development life cycle processes of ISO/IEC 12207:2008 and are presented herein in full compliance with ISO/IEC 24774:2010. The content of these three standards provides the foundation of this TR.

This TR does not address:

- areas already covered by existing related standards, e.g. the international standards that relate to the four standards used to build this TR (see Bibliography);
- FDA guidance documents; or
- software development tools.

This TR describes the PRM for medical device software development and is limited in scope to the life cycle processes described in IEC 62304:2006. The process names correspond to those of IEC 62304:2006. The mappings provided in Annex B are essential for the alignment between IEC 62304:2006 (which is based on ISO/IEC 12207:1995) and ISO/IEC 12207:2008, developed to address the detailed normative relationship between the two standards.

This technical report is not intended to be used as the basis of regulatory inspection or certification assessment activities.

2 Normative references

The following documents, in whole or in part, are normatively referenced in this document and are indispensable for its application. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC 62304:2006, *Medical device software – Software life cycle processes*

ISO/IEC 12207:2008, *Systems and software engineering – Software life cycle processes*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in IEC 62304:2006 apply.

NOTE To be consistent with the requirements for developing a PRM, the guidelines set forth in ISO/IEC 24774 were followed. Having a dedicated software risk management process enables the software developers to realize the set of requirements they have to adhere to when developing software for medical devices. This PRM also enables the medical device software developers to determine the requirements necessary to develop software for a specific safety class. The PRM presented in this TR includes only the software risk management requirements of ISO 14971 that are a part of IEC 62304. The software risk management terminology is therefore derived directly from ISO 14971. For the purposes of this TR, the software development-related terms and definitions used are inherited from IEC 62304.

4 Medical device software life cycle processes

4.1 Software development process

4.1.1 Software development planning

4.1.1.1 Purpose

The purpose of software development planning (IEC 62304, 5.1) is to establish a plan for conducting the activities of the software development processes.

4.1.1.2 Outcomes

The successful implementation of software development planning shall ensure that:

a) a software development plan is established for the software development appropriate to the scope, magnitude, and software safety classification of the software system. [Classes A, B, C];

NOTE 1 The software development plan includes the description of the development processes, the deliverables from the processes (including documentation), software configuration and change management (including SOUP configuration items and software used to support development), and software problem resolution.

b) the software development plan addresses how traceability between system requirements, software requirements, software system test and risk control measures is established [Classes A, B, C];

c) the software development plan is maintained throughout the software life cycle [Classes A, B, C];

d) the software development plan references system design and development [Classes A, B, C];

e) the software development plan includes or references the standards, methods and tools associated with the development of software items for Class C [Class C];

f) the software development plan includes or references an integration strategy for software units, including SOUP [Classes B, C];

g) the software development plan includes or references a verification strategy [Classes A, B, C];

NOTE 2 Verification strategy includes ensuring that all activities and tasks are complete along with all the associated documentation.

h) the software development plan includes or references a risk management plan, including the plan to manage risks relating to SOUP [Classes A, B, C];

i) the software development plan includes or references a strategy identifying the documentation to be produced during the software development life cycle, and the standards to be applied for the development of the software documentation [Classes A, B, C];

j) the software development plan includes or references a configuration management plan [Classes A, B, C];

NOTE 3 The software configuration management plan includes or references:

i) the classes, types, categories or lists of items to be controlled;

ii) the software configuration management activities and tasks;

iii) the organization(s) responsible for performing software configuration management and activities;

iv) their relationship with other organizations, such as software development or maintenance;

v) when the items are to be placed under configuration control;

vi) when the problem resolution process is to be used;

vii) software configuration items that include other software products or entities such as SOUP.

k) the software development plan includes or references the supporting items or settings used to develop medical device software requiring control [Classes B, C];

l) The software development plan includes the plan to place configuration items under documented configuration management control before they are verified [Classes B, C].

4.1.2 Software requirements analysis

4.1.2.1 Purpose

The purpose of software requirements analysis (IEC 62304, 5.2) is to establish the requirements of the software elements of the system.

4.1.2.2 Outcomes

A successful implementation of software requirements analysis shall ensure that:

a) the requirements allocated to the software system and their interfaces are defined [Classes A, B, C];

b) software requirements are analyzed for correctness and testability [Classes A, B, C];

c) the impact of software requirements on the operating environment are understood [Classes A, B, C];

d) consistency and traceability are established between the software requirements and system requirements [Classes A, B, C];

e) prioritization for implementing the software requirements is defined [ISO/IEC 12207];

f) the existing requirements, including system requirements, are updated as appropriate as a result of software requirements analysis [Classes A, B, C];

g) changes to the software requirements are evaluated for cost, schedule and technical impact [ISO/IEC 12207];

h) the software requirements are baselined and communicated to all affected parties [ISO/IEC 12207];

i) risk control measures implemented in software for hardware failures and potential software defects are included in the software requirements [Classes B, C];

NOTE 1 Software architecture implements the defined risk management requirements.

NOTE 2 Software safety class is assigned to software items based on the possible effects of the hazard.

j) medical device risk analysis is re-evaluated and updated as appropriate when software requirements are established [Classes A, B, C].

4.1.3 Software architectural design

4.1.3.1 Purpose

The purpose of software architectural design (IEC 62304, 5.3) is to provide a design for the software that implements and can be verified against the requirements.

4.1.3.2 Outcomes

A successful implementation of software architectural design shall ensure that:

a) a software architectural design is developed and baselined that describes the software items, including SOUP, that will implement the software requirements [Classes B, C];

b) in the case of SOUP items, all functional and performance requirements shall be specified, including hardware and software requirements of the system [Classes B, C];

NOTE 1 Examples include processor type and speed, memory type and size, system software type, communication and display software requirements.

c) internal and external interfaces of each software item are defined [Classes B, C];

d) consistency and traceability are established between software requirements and software design [ISO/IEC 12207];

e) the effectiveness of the segregation between the software items that is essential to risk control is identified and ensured [Class C];

NOTE 2 An example of segregation is to have software items execute on different processors. The effectiveness of the segregation can be ensured by having no shared resources between the processors.

f) software architecture implements system and software requirements, including those relating to risk control [Classes B, C].

4.1.4 Software detailed design

4.1.4.1 Purpose

The purpose of software detailed design (IEC 62304, 5.4) is to provide a design for the software detailed enough to permit coding and testing.

4.1.4.2 Outcomes

A successful implementation of software detailed design shall ensure that:

a) the software architecture is refined into software units [Classes B, C];

b) a detailed design of each software unit of the software item is developed [Class C];

c) external interfaces of each software unit are defined [Class C];

d) consistency and traceability are established between the detailed design and the requirements and architectural design [ISO/IEC 12207];

e) a detailed design is verified and documented to ensure that it implements and does not contradict the software architecture [Class C].

4.1.5 Software unit implementation and verification

4.1.5.1 Purpose

The purpose of software unit implementation and verification (IEC 62304, 5.5) is to produce executable software units that properly reflect the software design.

4.1.5.2 Outcomes

A successful implementation of software unit implementation and verification shall ensure that:

a) software units defined by the design are produced [Classes A, B, C];

NOTE 1 For Class A medical device software developers, it is not necessary to base software units on a software design.

b) verification criteria are defined for all software units against their requirements [Classes B, C];

NOTE 2 Where verification is done by testing, the test procedures are evaluated for correctness.

c) consistency and traceability are established between software units and requirements and design [ISO/IEC 12207];

d) software unit acceptance criteria prior to their integration into larger software items are established and it is ensured that software units mee the acceptance criteria [Classes B, C];

e) additional software unit acceptance criteria for Class C medical device software are established and it is ensured that Class C medical device software units mee the criteria [Class C];

f) verification of the software units against the requirements and the design is accomplished and documented [Classes B, C].

4.1.6 Software integration and integration testing

4.1.6.1 Purpose

The purpose of software integration and integration testing (IEC 62304, 5.6) is to combine the software units producing integrated software items, consistent with the software design, that demonstrate that the functional and non-functional software requirements are satisfied on an equivalent or complete operational platform.

4.1.6.2 Outcomes

A successful implementation of software integration and integration testing shall ensure that:

a) software items defined by the integration strategy are produced [Classes B, C];

b) software items are verified using the defined criteria [Classes B, C];

c) the hardware and software items and support for manual operation are integrated into the system [Classes B, C];

d) integrated software items are tested and the results of integration testing are recorded [Classes B, C];

NOTE 1 Examples to be considered in integration testing: i) the required functionality of the software, ii) implementation of risk control measures, iii) specified timing and other behaviour, iv) specified functioning of internal and external interfaces, and v) testing under abnormal conditions including foreseeable misuse.

NOTE 2 Integration test records permit the test repeatability including: i) the test results (pass/fail and a list on anomalies), and ii) identification of the tester.

e) verification criteria for software items are developed that ensure compliance with the software requirements allocated to the items [Classes B, C];

f) consistency and traceability are established between software design and software items [ISO/IEC 12207];

g) a regression strategy is developed and applied for re-verifying software items when a change in software units (including associated requirements, design and code) occur [Classes B, C];

h) anomalies found during software integration and integration testing are managed in accordance with the software problem resolution process [Classes B, C].

4.1.7 Software system testing

4.1.7.1 Purpose

The purpose of software system testing (IEC 62304, 5.7) is to confirm that the integrated software system meets its defined requirements.

4.1.7.2 Outcomes

A successful implementation of software system testing shall ensure that:

a) criteria for the integrated software is developed that demonstrates compliance with the software requirements [Classes B, C];

NOTE 1 A set of tests is established, expressed as input stimuli, expected outcomes, pass/fail criteria and procedures for conducting software qualification testing.

b) integrated software is verified using the defined criteria [Classes B, C];

NOTE 2 A set of tests is conducted for software system testing, such that all software requirements are covered.

c) anomalies found during the software system testing process are managed in accordance with the software problem resolution process [Classes B, C];

d) a regression strategy is developed and applied for retesting the integrated software when a change in software items is made [Classes B, C];

NOTE 3 A regression strategy can demonstrate that unintended side effects have not been introduced.

e) when changes to software items are made during software system testing, relevant risk management activities are performed [Classes B, C];

f) software system testing is verified [Classes B, C];

NOTE 4 Verification covers: i) the appropriateness of verification strategies and test procedures, ii) traceability of software system test procedures to software requirements, iii) coverage of all software requirements in verification testing, and iv) ensuring that the test results meet the required pass/fail criteria.

g) test results are recorded to permit test repeatability [Classes B, C].

NOTE 5 Software system test records contain the test results (pass/fail and a list of anomalies) and identification of the tester.

4.1.8 Software release

4.1.8.1 Purpose

The purpose of software release (IEC 62304, 5.8) is to confirm that each software work product and/or service of a process or project properly reflects the specified requirements.

4.1.8.2 Outcomes

A successful implementation of software release shall ensure that:

a) the completeness of software verification is ensured [Classes A, B, C];

NOTE 1 Verification activities include evaluation of the results prior to software release.

NOTE 2 When re-releasing software products (in the case of problem or modification resolution) software verification is required for all safety classifications [A, B, C].

b) known residual anomalies are identified and recorded [Classes B, C];

c) the evaluation of all known residual anomalies and their potential contribution to an unacceptable risk is ensured [Classes B, C];

d) the completeness of all activities and tasks along with the associated documentation is ensured [Classes B, C];

e) results of the verification activities are made available to the customer and other involved parties [ISO/IEC 12207].

4.2 Software maintenance

NOTE 1 The software maintenance process in this document is a specialization of the maintenance process of ISO/IEC 15288 [4]. Users can consider claiming conformance to the 15288 process rather than the process in ISO/IEC 12207:2008.

NOTE 2 The software maintenance process of this document is compatible with ISO/IEC 14764:2006 [2].

4.2.1 Purpose

The purpose of the software maintenance process (IEC 62304, Clause 6) is to provide cost-effective support to a delivered software product.

NOTE Pre-delivery software maintenance activities include planning for post-delivery operations, supportability, and logistics determination. Post-delivery activities include software modification and operational support, such as training or operating a help desk.

4.2.2 Outcomes

A successful implementation of the software maintenance process shall ensure that:

a) a maintenance strategy is developed to manage modification of products, including SOUP items, according to the release strategy [Classes A, B, C];

NOTE 1 Procedures for receiving, recording, evaluating, resolving and tracking problem reports and modification requests from the users are established.

NOTE 2 The use of a risk management process and software configuration management process for management changes and modifications is addressed.

NOTE 3 Procedures to evaluate and implement the upgrades, bug fixes, patches and obsolescence of SOUP are defined.

b) all feedback is monitored, documented and evaluated to ensure that system and software documentation is updated as needed [Classes A, B, C];

c) impacts of problem reports on safety are analysed [Classes A, B, C];

NOTE 4 Each problem report is evaluated to determine how it affects the safety of a released software system,

d) the impact of changes to the existing system on organization, operations or interfaces are identified [Classes B, C];

e) change requests that modify released software system, including the associated documentation, are evaluated and approved [Classes A, B, C];

f) the software system modification is communicated to all affected parties [Classes A, B, C];

NOTE 5 These modifications include the communication about the consequences of continued unchanged use of the released software as well as guidelines of installing the changes to the released software.

g) modified software is developed with associated tests that demonstrate that requirements are not compromised [Classes A, B, C];

NOTE 6 Requirements of software changes relating to safety are handed over to software risk management process.

h) software upgrades are migrated to the customer's environment [ISO/IEC 12207].

4.3 Software risk management

4.3.1 Purpose

The purpose of the software risk management process (IEC 62304, Clause 7) is to ensure that all hazards related to software are addressed.

4.3.2 Outcomes

A successful implementation of the software risk management process shall ensure that:

a) software items that could contribute to a hazardous situation are identified [Classes B, C];

NOTE 1 The hazardous situation could be the direct result of software failure or the result of the failure of a risk control measure that is implemented in software.

b) potential causes of contribution to a hazardous situation are identified [Classes B, C];

NOTE 2 Potential causes include: a) incorrect or incomplete specification of functionality, b) software defects in the identified software item functionality, c) failure or unexpected results from SOUP, d) hardware failures or other software defects that could result in unpredictable software operation, and e) reasonably foreseeable misuse.

c) published SOUP anomaly lists are evaluated [Classes B, C];

NOTE 3 The SOUP anomaly lists are evaluated to determine if any of the anomalies result in a sequence of events that could result in a hazardous situation.

d) potential causes of contribution to a hazardous situation are documented [Classes B, C];

NOTE 4 The potential causes are documented in a risk management file.

e) sequences of events that could result in a hazardous situation are documented [Classes B, C];

NOTE 5 The potential causes are documented in a risk management file.

f) risk control measures are defined for each documented potential cause of software item contributing to a hazardous situation [Classes B, C];

NOTE 6 The risk control measures can be implemented in hardware, software, the working environment or user instruction.

g) risk control measures that are implemented as functions of software items are included in software requirements and a software safety class is assigned to the software items [Classes B, C];

NOTE 7 The software safety class of the software item is assigned based on the possible effects of the hazard that the risk control measure is controlling.

NOTE 8 The software item is developed in accordance with the software development processes.

h) the implemented risk control measures are verified and the verification is documented [Classes B, C];

i) the implemented risk control measures are evaluated to identify any new sequences of events that could result in a hazardous situation [Classes B, C];

NOTE 9 The identified new sequences of events are documented in the risk management file.

j) the traceability of software hazards is established and documented [Classes B, C];

NOTE 10 The traceability from a) the hazardous situation to the software item, b) the software item to the specific software cause, c) the software cause to the risk control measure, and d) the risk control measure to the verification of the risk control measure.

k) changes to medical device software are analyzed to determine if additional potential causes contributing to a hazardous situation that require additional software risk control measures are introduced [Classes A, B, C];

l) the impact of software changes, including changes to SOUP, on existing risk control measures is analysed [Classes B, C];

m) relevant risk management activities are performed based on the impact analyses of the software changes [Classes B, C].

4.4 Software configuration management

NOTE The software configuration management process is a specialization of the configuration management process from the project process group in ISO/IEC 12207:2008.

4.4.1 Purpose

The purpose of the software configuration management process (IEC 62304, Clause 8) is to establish and maintain the integrity of the software items of a process or project and make them available to concerned parties.

4.4.2 Outcomes

A successful implementation of the software configuration management process shall ensure that:

a) items generated by the process or project are identified, defined and documented [Classes A ,B, C];

NOTE 1 In case of a SOUP configuration item, its details are documented, including the title, the developer and the unique SOUP designator of each SOUP configuration item used.

b) the set of configuration items and their versions that comprise the software system configuration are documented [Classes A, B, C];

c) modifications to the configuration items are implemented and the releases of the items are controlled [Classes A ,B, C];

NOTE 2 Changes to configuration items are made only in response to approved change requests while modifications to the existing system are managed through the software configuration management process.

NOTE 3 Modified software items are implemented and verified; and an audit trail exists, whereby each modification, the reason for the modification, and authorization of the modification can be traced.

NOTE 4 Activities that need to be repeated as a result of the change, including changes to the software safety classification of software system and software items, are identified and performed.

d) modifications and releases are made available to affected parties [ISO/IEC 12207];

e) the status of the items and modifications are recorded [Classes A ,B, C];

NOTE 5 The records of controlled configuration items include system configuration.

f) the completeness and consistency of the items is ensured [ISO/IEC 12207];

g) the storage, handling and delivery of the items are controlled [Classes B, C].

NOTE 6 The procedures and the environment used to create released software are documented.

NOTE 7 The documentation archiving time is determined as being the longer of: the lifetime of the device or the time specified by the relevant regulatory requirements.

NOTE 8 The version of the released software is documented.

4.5 Software problem resolution

4.5.1 Purpose

The purpose of the software problem resolution process (IEC 62304, Clause 9) is to ensure that all discovered problems are identified, analyzed, managed and controlled to resolution.

4.5.2 Outcomes

A successful implementation of the software problem resolution process:

a) problems are recorded, identified and classified based on the type, scope and criticality [Classes A, B, C];

NOTE 1 Problem reports include actual or potential adverse events, and deviations from specifications.

NOTE 2 Any change of safety class in the software system or its software items is identified.

b) problems are analyzed and assessed to identify acceptable solution(s), and the records of problem analysis and assessment are maintained [Classes A, B, C];

NOTE 3 Change requests for actions needed to correct identified problems are created, or the rationale for taking no action is documented.

c) problem's relevance to safety is evaluated using the software risk management process [Classes A, B, C];

NOTE 4 The outcome of the evaluation is documented and a change request for action needed to correct the problem is created, or else the rationale for taking no action is documented.

d) relevant parties are advised of the existence of the problem as appropriate [Classes A, B, C];

e) problem resolution is implemented in accordance with the software configuration management and software system testing processes [Classes A, B, C];

NOTE 5 The results of the retesting include:

i) test results;

ii) anomalies found;

iii) the version of software tested;

iv) relevant hardware and software test configurations;

v) relevant test tools;

vi) date tested; and

vii) identification of the tester.

f) the records of problem reports, problem resolutions and their verification are maintained [Classes A, B, C];

NOTE 6 The risk management file is updated if necessary.

g) the status of all problems reported and trends across problems are known [Classes A, B, C];

NOTE 7 The software problem resolution process could be used or easily adapted to manage, track and control software change requests.

h) problems are tracked to closure in accordance with the software configuration management process [Classes A, B, C];

NOTE 8 Adverse trends are reversed.

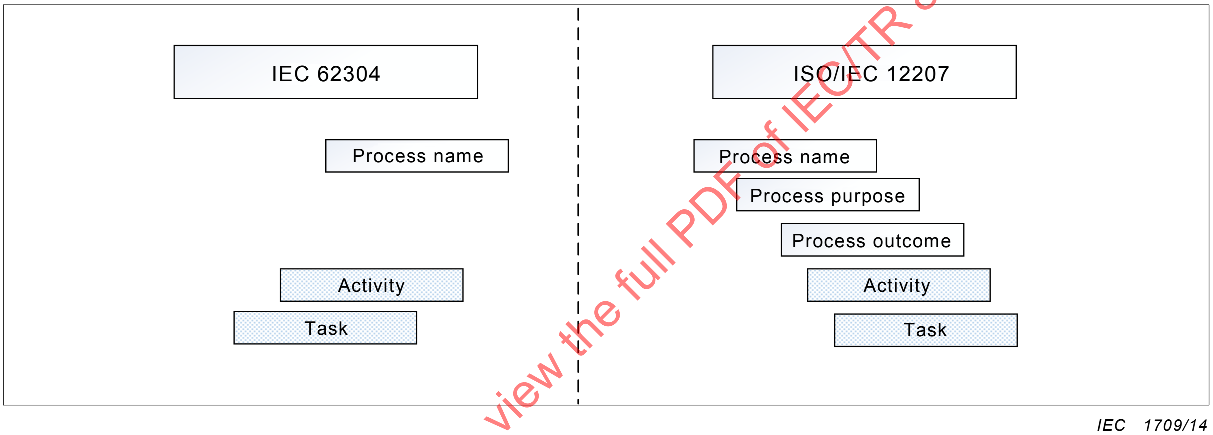
i) the records of testing, retesting or regression testing after software change are retained [Classes A, B, C].

NOTE 9 The test documentation includes: i) test results, ii) anomalies found, iii) the version of the software tested, iv) relevant hardware and software test configurations, v) relevant test tools, vi) date tested, and vii) identification of the tester.

Annex A
(informative)

Development of this technical report

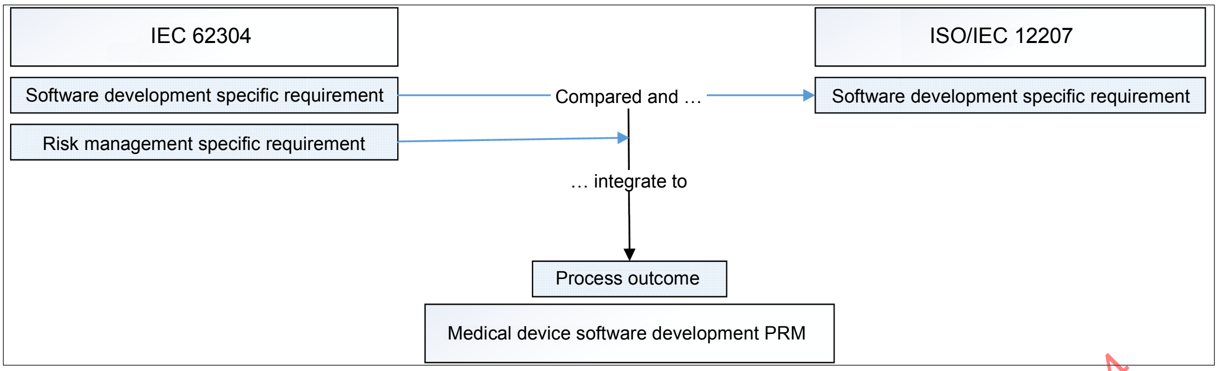
Figure A.1 illustrates how the two standards used for building the PRM for medical device software development (IEC 62304:2006 and ISO/IEC 12207:2008) present requirements within their process descriptions (the process elements containing the requirement are dotted in Figure A.1). The process requirements in IEC 62304:2006 are presented at the activity level. IEC 62304:2006 does not provide process descriptions through a process purpose statement. In ISO/IEC 12207:2008, requirements are described on both activity and/or task levels after the purpose and the outcomes of the process are described. In both of these standards, requirements described in one activity or task can contain the development of many various results. Based on ISO/IEC 24774, the process outcomes should be one sentence statements focusing only on one requirement at a time. Process outcomes are in a logical sequence of activities in which these requirements could be achieved in a software development life cycle.



IEC 1709/14

Figure A.1 – Requirements in process elements of IEC 62304:2006 and ISO/IEC 12207:2008

In the PRM for the medical device software development, the software development process requirements in IEC 62304:2006 were mapped against the process outcomes of ISO/IEC 12207:2008. If a corresponding outcome existed that mapped to an IEC 62304 requirement, it was adopted in the PRM for medical device software development together with its safety class. In the case of a process outcome without a corresponding safety class, the process outcome is derived from ISO/IEC 12207:2008 without a corresponding requirement in IEC 62304:2006. The informal and exemplary information of IEC 62304:2006 was incorporated into the PRM in the form of notes related to the corresponding process outcome.


IEC 62304
Software development specific requirement
Risk management specific requirement
Compared and …
… integrate to
ISO/IEC 12207
Software development specific requirement
Process outcome
Medical device software development PRM

IEC 1710/14

Figure A.2 – Development of process outcomes for medical device software development PRM

The scope of the PRM is limited to the processes of IEC 62304. Table A.1 illustrates the 10 IEC 62304 processes (out of the 12) that map directly to the processes in ISO/IEC 12207:2008. The remaining two processes, software development planning and software risk management, do not directly map to ISO/IEC 12207.

Table A.1 – Direct process mappings between IEC 62304:2006 and ISO/IEC 12207:2008

IEC 62304 process		Corresponding ISO/IEC 12207:2008 process
5.2	Software requirements analysis	Software requirements analysis
5.3	Software architectural design	Software architectural design
5.4	Software detailed design	Software detailed design
5.5	Software unit implementation and verification	Software construction
5.6	Software integration and integration testing	Software integration
5.7	Software system testing	Software qualification testing
5.8	Software release	Software verification
6	Software maintenance	Software maintenance
8	Software configuration management	Software configuration management
9	Software problem resolution	Software problem resolution

Annex B
(informative)

Mapping between IEC 62304:2006 and ISO/IEC 12207:2008

The mapping of the requirements from two different international standards aims at integrating the varying underlying requirements into a more abstract set of PRM-based requirements which can be applied in the development of a medical device software development PRM.

With the exception of the software risk management process, the majority of the IEC 62304 processes are mapped to their ISO/IEC 12207:2008 counterparts. In conducting process mappings for the directly corresponding processes, the systematic approach of constant comparison and memoing as described by the grounded theory method was applied. Constant comparison is an iterative process of data integration where the dimensions and the properties specific to data are specified. Several iterations of constant comparison and memoing were conducted before the final mapping of each process outcome was agreed upon.

Table B.1 presents the mapping results between process outcomes of ISO/IEC 12207:2008 and IEC 62304:2006. The first two columns from the left contain the process names and subclause numbers of IEC 62304 requirements respectively. The list of the sequential process outcomes derived from these requirements are shown in the third column of the table. The safety classes that are related to the process outcomes are shown in the next three columns to right of the outcomes. Providing safety class(es) for each process outcome helps medical device software developers to identify the set of requirements that apply specifically to the safety class of their software. The corresponding ISO/IEC 12207 outcome numbers and process names are shown in the seventh and eighth columns, respectively.

Table B.1 – Mapping between process outcomes of the PRM and the requirements of IEC 62304:2006, including their safety classes, and the requirements of ISO/IEC 12207:2008 *(1 of 9)*

IEC 62304 process	IEC 62304 subclause	Medical device software development PRM process outcomes	Safety class			ISO/IEC 12207 subclause	ISO/IEC 12207 process
			A	B	C		
5.1 Software development planning	5.1.1 (a), (b), (d), (e)	(a) a software development plan is established for the software development appropriate to the scope, magnitude, and software safety classification of the software system	✓	✓	✓	7.1.1.2 a)	Software implementation (7.1)
	5.1.1 (c)	(b) the software development plan addresses how traceability between system requirements, software requirements, software system test and risk control measures is established	✓	✓	✓		
	5.1.2	(c) the software development plan is maintained throughout the software life cycle	✓	✓	✓		
	5.1.3	(d) the software development plan references system design and development	✓	✓	✓		
	5.1.4	(e) the software development plan includes or references the standards, methods and tools associated with the development of software items for Class C			✓		
	5.1.5	(f) the software development plan includes or references an integration strategy for software units, including SOUP		✓	✓	7.1.6.2 a)	
	5.1.6	(g) the software development plan includes or references a verification strategy	✓	✓	✓	7.2.4.2 a), b)	
	5.1.7	(h) the software development plan includes or references a risk management plan, including the plan to manage risks relating to SOUP	✓	✓	✓		
	5.1.8	(i) the software development plan includes or references a strategy identifying the documentation to be produced during the software development life cycle, and the standards to be applied for the development of the software documentation	✓	✓	✓	7.2.1.2 a), b)	
	5.1.9	(j) the software development plan includes or references a configuration management plan	✓	✓	✓	7.2.2.2 a)	
	5.1.10	(k) the software development plan includes or references the supporting items or settings used to develop medical device software requiring control		✓	✓		
	5.1.11	(l) the software development plan includes the plan to place configuration items under documented configuration management control before they are verified		✓	✓		

Table B.1 *(2 of 9)*

IEC 62304 process	IEC 62304 subclause	Medical device software development PRM process outcomes	Safety class			ISO/IEC 12207 subclause	ISO/IEC 12207 process
			A	B	C		
5.2 Software requirements analysis	5.2.1	(a) the requirements allocated to the software system and their interfaces are defined	✓	✓	✓	7.1.2.2 a)	Software requirements analysis (7.1.2)
	5.2.6 (a)-(e)	(b) software requirements are analyzed for correctness and testability	✓	✓	✓	7.1.2.2 b)	
	5.2.2	(c) the impact of software requirements on the operating environment are understood	✓	✓	✓	7.1.2.2 c)	
	5.2.6 (f)	(d) consistency and traceability are established between the software requirements and system requirements	✓	✓	✓	7.1.2.2 d)	
		(e) prioritization for implementing the software requirements is defined				7.1.2.2 e)	
	5.2.5	(f) the existing requirements, including system requirements, are updated as appropriate as a result of software requirements analysis	✓	✓	✓	7.1.2.2 f)	
		(g) changes to the software requirements are evaluated for cost, schedule and technical impact				7.1.2.2 g)	
		(h) the software requirements are baselined and communicated to all affected parties				7.1.2.2 h)	
	5.2.3	i) risk control measures implemented in software for hardware failures and potential software defects are included in the software requirements		✓	✓		
	5.2.4	j) medical device risk analysis is re-evaluated and updated as appropriate when software requirements are established	✓	✓	✓		

Table B.1 *(3 of 9)*

IEC 62304 process	IEC 62304 subclause	Medical device software development PRM process outcomes	Safety class			ISO/IEC 12207 subclause	ISO/IEC 12207 process
			A	B	C		
5.3 Software architectural design	5.3.1; 5.3.6 (a) (b) (c)	(a) a software architectural design is developed and baselined that describes the software items, including SOUP, that will implement the software requirements		✓	✓	7.1.3.2 a)	Software architectural design (7.1.3)
	5.3.3, 5.3.4	(b) In the case of SOUP items, all functional and performance requirements shall be specified, including hardware and software requirements of the system		✓	✓		
	5.3.2	(c) internal and external interfaces of each software item are defined		✓	✓	7.1.3.2 b)	
		(d) consistency and traceability are established between software requirements and software design				7.1.3.2 c)	
	5.3.5	(e) Identify and ensure the effectiveness of the segregation between the software items that is essential to risk control.			✓		
	5.3.6 (a)	(f) Ensure that software architecture implements system and software requirements including those relating to risk control.		✓	✓		
5.4 Software detailed design	5.4.1	(a) software architecture is refined into software units		✓	✓		Software detailed design (7.1.4)
	5.4.2	(b) a detailed design of each software unit of the software item is developed			✓	7.1.4.2 a)	
	5.4.3	(c) external interfaces of each software unit are defined			✓	7.1.4.2 b)	
		(d) consistency and traceability are established between the detailed design and the requirements and architectural design				7.1.4.2 c)	
	5.4.4	(e) a detailed design is verified and documented to ensure that it implements and does not contradict the software architecture			✓		

Table B.1 *(4 of 9)*

IEC 62304 process	IEC 62304 subclause	Medical device software development PRM process outcomes	Safety class			ISO/IEC 12207 subclause	ISO/IEC 12207 process
			A	B	C		
5.5 Software unit implementation and verification	5.5.1	(a) software units defined by the design are produced	✓	✓	✓	7.1.5.2 b)	Software construction (7.1.5)
	5.5.2	(b) verification criteria are defined for all software units against their requirements		✓	✓	7.1.5.2 a)	
		(c) consistency and traceability are established between software units and requirements and design				7.1.5.2 c)	
	5.5.3	(d) software unit acceptance criteria prior to their integration into larger software items is established and software units meeting the acceptance criteria is ensured		✓	✓		
	5.5.4	(e) additional software unit acceptance criteria for Class C medical device software are established and the Class C medical device software units meeting the criteria are ensured			✓		
	5.5.5	(f) verification of the software units against the requirements and the design is accomplished and documented		✓	✓	7.1.5.2 d)	

Table B.1 *(5 of 9)*

IEC 62304 process	IEC 62304 subclause	Medical device software development PRM process outcomes	Safety class			ISO/IEC 12207 subclause	ISO/IEC 12207 process
			A	B	C		
5.6 Software integration and integration testing	5.6.1	(a) software items defined by the integration strategy are produced		✓	✓	7.1.6.2 d)	Software integration (7.1.6)
	5.6.2 (a)	(b) software items are verified using the defined criteria		✓	✓	7.1.6.2 c)	
	5.6.2 (b)	(c) the hardware and software items and support for manual operation are integrated into the system		✓	✓		
	5.6.3	(d) integrated software items are tested and the results of integration testing are recorded		✓	✓	7.1.6.2 e)	
	5.6.4	NOTE1 Examples to be considered in integration testing: i) the required functionality of the software, ii) implementation of risk control measures, iii) specified timing and other behaviour, iv) specified functioning of internal and external interfaces, and v) testing under abnormal conditions including foreseeable misuse					
	5.6.7	NOTE2 Integration test records permit the test repeatability including: i) the test results (pass/fail and a list on anomalies), and ii) identification of the tester.					
	5.6.5	(e) verification criteria for software items are developed that ensure compliance with the software requirements allocated to the items		✓	✓	7.1.6.2 b)	
		(f) consistency and traceability are established between software design and software items				7.1.6.2 f)	
	5.6.6	(g) a regression strategy is developed and applied for re-verifying software items when a change in software units (including associated requirements, design and code) occur		✓	✓	7.1.6.2 g)	
	5.6.8	h) anomalies found during software integration and integration testing are managed in accordance with software problem resolution process		✓	✓		