Study on safety of medical devices software

Conducted by Serma Ingenierie at the request of the ANSM
(French National Agency for Medicines and Health Products Safety)

July 2016

TABLE OF CONTENTS

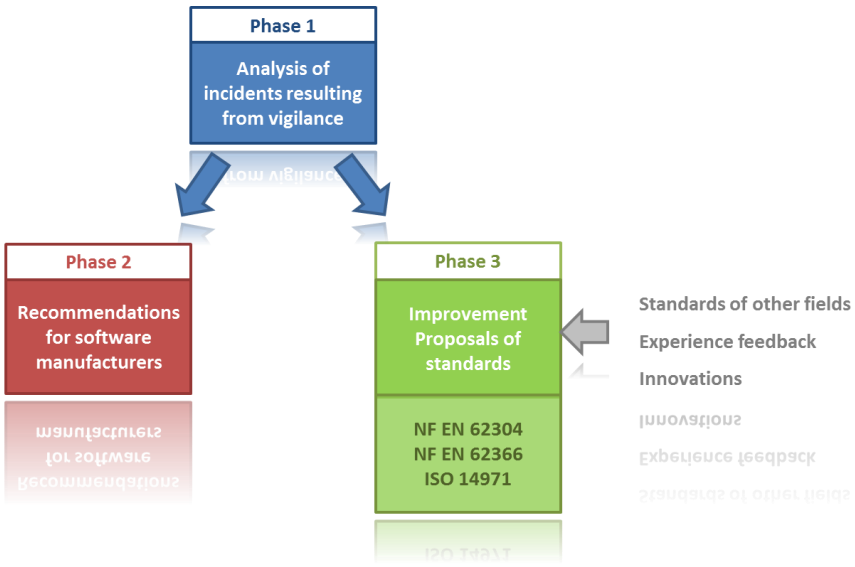
SUMMARY

To meet the growing importance of software in medical applications, ANSM (French National Agency for Medicines and Health Products Safety) launched a study on safety of software. This study was carried out between August 2014 and November 2015 by SERMA INGENIERIE. Main objectives were to:

- complete European thoughts on software safety at a normative level,
- assess relevance and adequacy of normative environment available for software developpement,
- produce recommendations about standards application for software manufacturers.

The chosen analysis process for this study included three major phases and was based on:

- incidents resulting from vigilance (materiovigilance, reactovigilance, pharmacovigilance),
- existing reference sources of medical field,
- normative state of the art in other fields,
- feedback experience of SERMA INGENIERIE and ANSM,
- medical innovations.



Main **recommendations** for DM software manufacturers (Phase 2) detailed in this report are related to:

- Databases,
- Verification of Specification / Architectural Design,
- Change impact analysis (in the software),
- Risks management and monitoring.

Improvement Proposals of standards (Phase 3) have also been formulated. These proposals are focused on existing standards (NF EN 62304, EN 62366, ISO 14971). Two types of improvements are proposed:

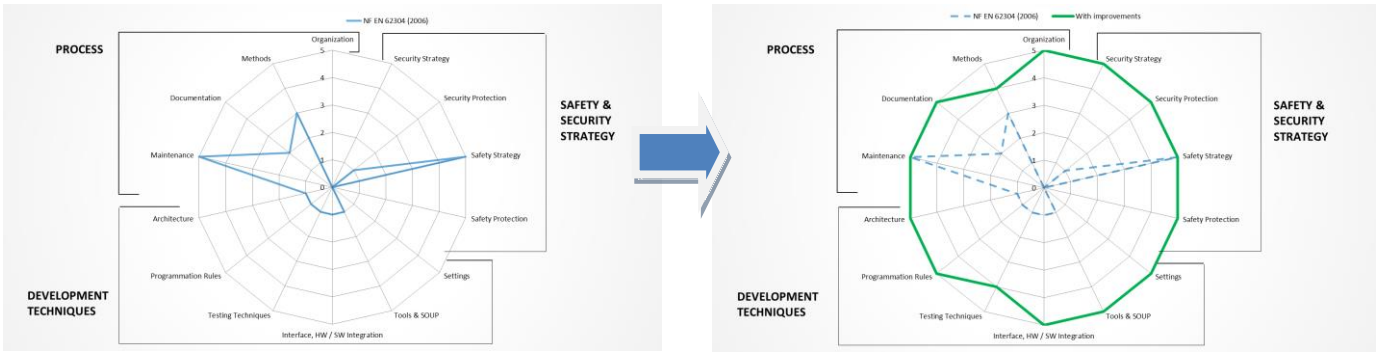
- Add precisions to requirements of current standards,
- Create new requirements.

Main improvement proposals are:

- For NF EN 62304 standard (Medical device software - Software life-cycle processes):

Process	- Organizational structure, independence of teams, and stakeholders skills - Process definition and assessment strategy - Precisions to provide on detailed design activity (Software Unit - UL)
Development techniques	- Addition of software interfaces and software integration requirements - Addition of software configured by application data requirements - Definition of architectural constraints - Addition of techniques introduced in tables - Definition of programming rules and code verification - Presentation of testing techniques - Clarification on unit testing activities - Addition of tools qualification (used in software developpement)
Safety and Security	- Definition of Software Security strategy - Definition of Software Security techniques

Standard improvement coverage[1] can be symbolized by the graphs below:



- For NF EN 62366 standard (Medical devices – Application of usability engineering to medical devices):

  - Improvement of users training requirements
  - Addition of precisions on the operating manual

- For ISO 14971 standard (Medical devices — Application of risk management to medical devices):

  - Addition of Risks Management Security activities

All recommendations and improvements given in this report were established to provide a better control of safety for software medical device. Recommendations also consider impact of future medical innovations.

[1] For criteria and notes indicated on the radar diagram, see Annex 2 in this report

1. ANSM PRESENTATION

ANSM - French National Agency for Medicines and Health Products Safety - has been created by the Law of 29 December 2011 related to strengthening of the sanitary safety of the medicine and the health products. It was set up on 1st May 2012.

ANSM is responsible for assessing benefits and risks associated to the use of health products throughout their lifecycle.

Within ANSM, DMDPT (working group for diagnostic medical devices and technical platforms) is in charge of:

- diagnostic medical devices,
- radiotherapy medical devices,
- software,
- technical platforms (anesthesiology / reanimation devices, functional substitution, surgical unit).

In addition to its work on the assessment of the benefits and risks associated with the use of health products within its field of competence, DMDPT ensures production and scientific/technical monitoring of scientific documentation. DMDPT also represents ANSM involved in European and International activities in line with strategic focus of ANSM.

2.PURPOSE OF THE DOCUMENT

2.1. INTRODUCTION

This document is the final report of the study performed by SERMA INGENIERIE on safety of medical device software (ANSM market of 04/08/2014 No. 2014C029) between August 2014 and November 2015.To meet the growing importance of software in medical applications, ANSM launched a study on safety of medical device software including:

- medical devices software,
- software used in medical biology laboratories which enter now within the agency's jurisdiction,
- software designed to help medicine prescribing for which ANSM receives alerts.

Indeed, software are increasingly present in the field of medical devices, in terms of autonomous software, which have the status of medical device but also in terms of embedded software in medical devices.
Main objectives of this study are:

- to complete European thoughts on software safety at regulatory and normative levels from experience of other industrial fields using specific software,
- to measure relevance and adequacy of the normative environment available to publishers of such software, especially based on experience feedback from incident statements made through medical device vigilance activities (materiovigilance, reactovigilance, pharmacovigilance),
- to make recommendations on standards application to software manufacturers.

2.2. REFERENCE DOCUMENTS

Reference	Name	Designation
[IEC 61508]	IEC 61508 (2011)	Functional safety of electrical/electronic/programmable electronic safety-related systems
[ISO 14971]	NF EN ISO 14971 (2007)	Medical devices — Application of risk management to medical devices
[TR 80002-1]	Technical Report related to 14971 IEC/TR 80002-1:2009	Guidance on the application of ISO 14971 to medical device software (Part 1).
[NF EN 60601-1]	NF EN 60601-1 (2014)	Medical electrical equipment - Part 1: General requirements for basic safety and essential performance
[ISO 13485]	ISO 13485 (2012)	Medical devices — Quality management systems — Requirements for regulatory purposes
[93/42/CEE]	93/42/CEE (1993)	Council Directive concerning medical devices
[NF EN 62304]	NF EN 62304 (2006)	Medical device software - Software life-cycle processes
[NF EN 62366]	NF EN 62366 (2008)	Medical devices – Application of usability engineering to medical devices
[EN 50128]	EN 50128 (2011)	Railway applications - Communication, signalling and processing systems - Software for railway control and protection systems
[ISO 26262]	ISO 26262 (2011)	Road vehicles - Functional safety
[IEC 60880]	IEC 60880 (2006)	Nuclear power plants – Instrumentation and control systems important to safety – Software aspects for computer-based systems performing category A functions
[EN 50159]	EN 50159 (2011)	Railway applications - Communication, signalling and processing systems - Safety-related communication in transmission systems
[FDA_Cybersecurity]	FDA Cybersecurity 02 october 2014	Content of Premarket Submissions for Management of Cybersecurity in Medical Devices

2.3. ABBREVIATIONS

Acronym	Definition
ADAS	Advanced Driver Assistance Systems
FMEA	Failure Mode and Effects Analysis
FMECA	Failure Modes, Effects and Criticality Analysis
ANSM	French National Agency for Medicines and Health Products Safety (Agence Nationale de Sécurité du Médicament et des produits de santé)
PHA	Preliminary Hazard Analysis
ASIL	Automotive Safety Integrity Level
COTS	Commercial Off-the-Shelf
DBMS	DataBase Management System
MD	Medical Device
EL	Software Item (Elément Logiciel) [NF EN 62304]: any identifiable part of a computer program [ISO / IEC 90003: 2004, definition 3.14, modified]
HW	HardWare
HMI	Human Machine Interface
IMA	software of medical imagery (logiciel d'IMAgerie médicale)
LABM	medical biology automaton software (Logiciel d'Automates de Biologie Médicale)
LAP	software for medical prescription (Logiciel d'Aide à la Prescription)
MMR	Risk control measures (Mesures de Maîtrise des Risques)
REX	Feedback experience (Retour d'EXpérience)
RT	RadioTherapy software
SILBM	computer system for medical biology laboratory (Système Informatique de Laboratoire de Biologie Médicale)
SSIL	Software Safety Integrity Level
SOUP	Software Of Unknown Provenance
SW	SoftWare
TU	Unit Testing (Tests Unitaires)
TI	Integration Testing (Tests d'Intégration)
UL	Software Unit (Unité Logicielle) [NF EN 62304]: software item that is not subdivided into other items

2.4. TERMINOLOGY

<table>
<tr><th>Terms</th><th>Definitions / Precisions</th></tr>
<tr><td>Activities (development phases)</td><td>Group of one or more interrelated or interactive tasks - Source [NF EN 62304]</td></tr>
<tr><td>Category</td><td>Corresponds to "Category" of belonging of DM: this information defines destination of the use of the software (ex.: radiotherapy software, software of Medical Imagery...)</td></tr>
<tr><td>Software Safety class</td><td>Software safety class shall be assigned by the manufacturer regarding their potential effects on the patient, the operator or other actors, resulting from a hazardous phenomenon to which software system can contribute.

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
(Source : [NF EN 62304])</td></tr>
<tr><td>Comparison between COTS and SOUP</td><td>
<table>
<tr><th></th><th>SOUP</th><th>COTS</th></tr>
<tr><td>Common characteristics</td><td colspan="2">Software element or software:
• Already developed and generaly available for use,
• Reduce the development time,
• Commercially available,
• Programming code not necessary available (impact on the maintainability, risks of "bugs")</td></tr>
<tr><td rowspan="3">Specific characteristics</td><td>• Not developed to be integrated in the developed system</td><td>• Defined by the market needs</td></tr>
<tr><td>• Doesn't meet the realisation processes of the applicable standard</td><td>• Relevance to the needs demonstrated by a wide range of users</td></tr>
<tr><td>• Recording of the development processes insufficient or not available</td><td>• Have a documentation contractually accessible and complete</td></tr>
</table>
</td></tr>
<tr><td>Incident</td><td>Reported incident, detected on a device during operation at the declarant, or during tests at the manufacturer</td></tr>
<tr><td>Generic Software</td><td>Software that can be used for a variety of installations by the provision of application-specific data and/or algorithms - Source [EN 50128]</td></tr>
<tr><td>Pre-existing software</td><td>Software developed prior to the application, including COTS (commercial off-the shelf) and open source software - Source [EN 50128]</td></tr>
<tr><td>Process</td><td>A set of interrelated or interacting activities that transform inputs into outputs - Source [NF EN 62304]</td></tr>
<tr><td>Risk</td><td>Combination of the probability of occurrence of harm and its severity - [ISO / IEC Guide 51:1999, definition 3.2]</td></tr>
</table>

Terms	Definitions / Precisions
Safety	Ensure that the software was designed to prevent random and involuntary risks. According to [ISO / IEC Guide 51: 1999, definition 3.1]: freedom from risk which is not tolerable
Hazardous situations	Circumstance in which people, property or the environment is/are exposed to one or more hazards [ISO / IEC Guide 51: 1999, definition 3.6] NOTE: See Appendix E of [ISO 14971] standard for an explanation of the relationship between "hazard" and "harzardous situation."
Security	Ensure that the software was designed to withstand any malicious attacks. According to [ISO / IEC 12207: 1995, definition 3.25]: Protection of information and data in order that unauthorized persons or systems cannot read or modify them and authorized persons or systems are not denied access to them
Tasks	A single piece of work that needs to be done - Source [NF EN 62304]

3.OVERVIEW OF THE STUDY

3.1. METHODOLOGY

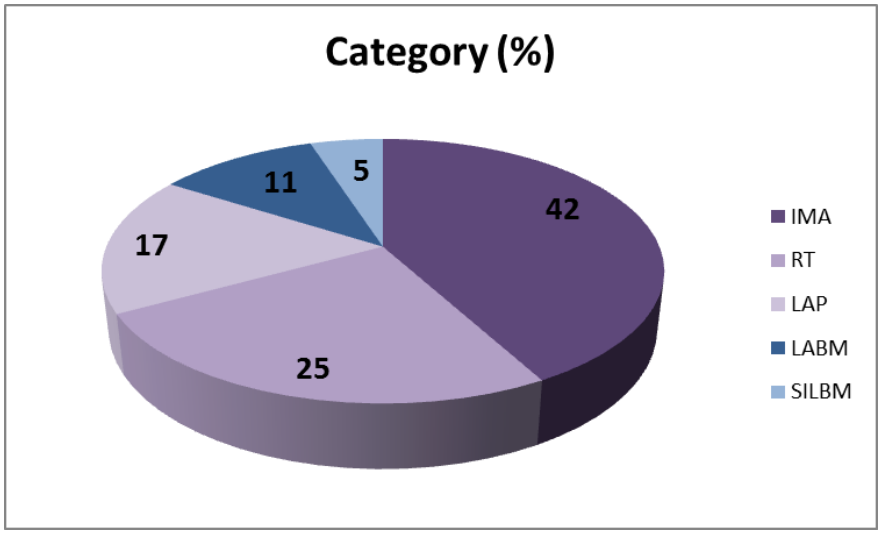
This study is composed of the following 3 phases:
- Phase 1: Performing an analysis of incidents provided by ANSM (inventory)
- Phase 2: Recommendations proposals for manufacturers
- Phase 3: Improvement areas of regulations and standards

3.1.1. PHASE 1: INVENTORY

Monitoring set up by the authorities as part of vigilances (including materiovigilance, reactovigilance, pharmacovigilance) allowed to raise various incidents that have been reported and grouped by ANSM.

All incidents concern operating software. Some incidents are directly raised by users and others by manufacturers.Manufacturers may have identified incidents after operators reporting, or after observation of an abnormal behavior during internal testing, performed after software has been placed on the market.

Based on a sample of 156 incidents proposed by ANSM, SERMA INGENIERIE has conducted a detailed analysis. This analysis focused on following categories:



Analysed incidents have been classified into two groups:
- Incidents related to non-compliance of standards.
- Incidents related to non-completeness of standards.

These information were used to perform phases 2 and 3.

3.1.2. PHASE 2: RECOMMENDATIONS PROPOSALS FOR MANUFACTURERS

First category of incidents identified in Phase 1 has helped to develop recommendations to medical device manufacturers to facilitate understanding and implementation of [NF EN 62304], [ISO 14971] and [NF EN 62366].

A synthesis of these recommendations is presented in chapter 4.1. Detail of these recommendations has been grouped as sheets included in this report in Chapter 5.2.

3.1.3. PHASE 3: IMPROVEMENT AXES OF REGULATIONS AND STANDARDS

Improvement proposals of [NF EN 62304], [ISO 14971] and [EN 62366] were identified from the analysis of three standards applied todifferent fields of application. In addition, the second category of incidents identified during Phase 1 (non-completeness of Standards) helped to strengthen these proposals.

A detailed analysis of eight standards from different fields of application and related to the development of critical software was performed.

The following standards coming from different industrial fields have been used for the studdy:

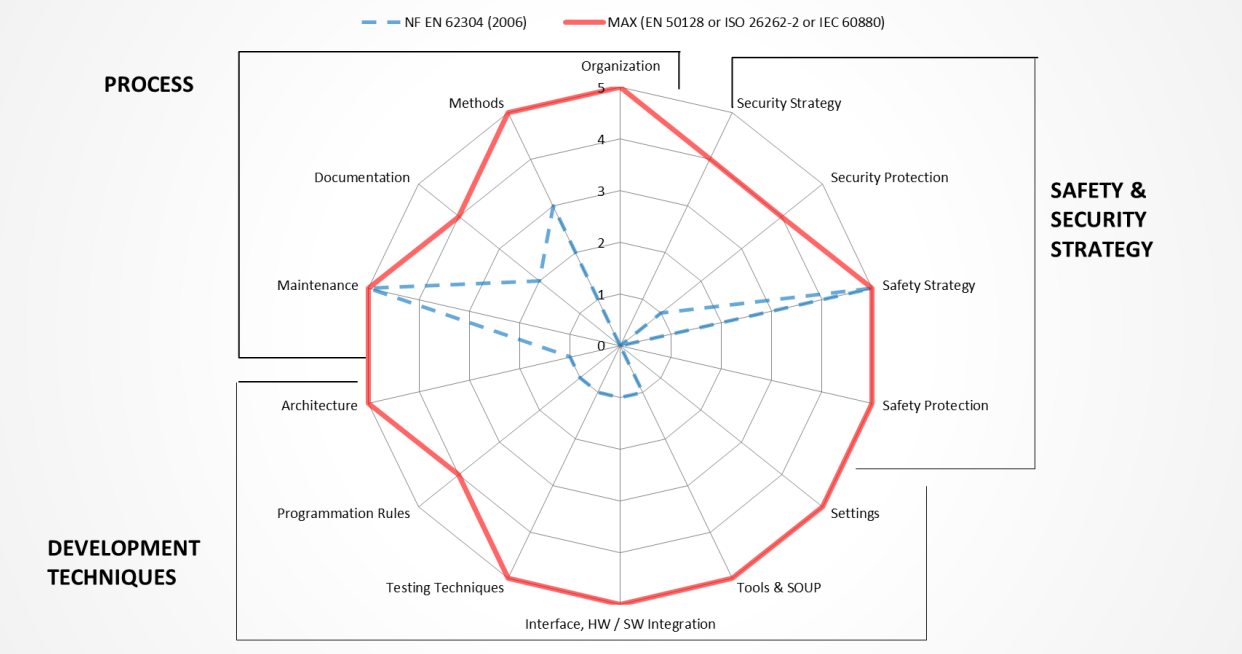
Standard	Date	Field
EN 50128	2011	Railway
ISO 26262-6	2011	Automotive
IEC 60880	2006	Nuclear
DO-178C	2012	Aerospace
IEC 62138	2009	Nuclear
IEC 61511	2003	Process
IEC 62061	2005	Machinery
ECSS-Q-ST-80-C	2009	Space

A multi-criteria study was able to highlight potential contributions of each reference source as compared to the current standard [NF EN 62304].

Analyzed criteria were classified into 3 categories as described below:

Categories	Criteria
Process	Organization
	Methods
	Documentation
	Maintenance
Development Techniques	Architecture
	Programming rules
	Testing Techniques
	Interface, HW / SW integration
	Tools & SOUP
	Settings
Safety & Security Strategy	Safety Protection
	Security Protection
	Safety Strategy
	Security Strategy

The analysis performed highlighted contribution of [EN 50128], [ISO 26262] and [IEC 60880] in this approach. A summary presentation of these standards is provided in Appendix 1 of this report. Their contribution is symbolized on the graph below:



A description of criteria and notes used in this radar diagram is given in Appendix 2 of this report.

[EN 50128] (Railway) is the standard which provides the most details about process aspects (Organization, Methods, Materials, Settings...) and a good level regarding to development techniques.

[ISO 26262-6] (Automotive) allows to complement development techniques including aspects regarding to architecture and hardware / software interface.

[IEC 60880] (Nuclear) was chosen because this standard incorporates concept of Security.Measures are required to take into account possible intentional external attacks, and ensure protection of information. In the medical context, this point is particularly not negligible towards the issue of medical privacy and malicious actions leading to a risk for the patient. To complete and clarify some points, consideration of provisions from [FDA_Cybersecurity] has also been studied.

The choosen strategy for analysis can be summarized as follows:

1. Comparison with [EN 50128] and [ISO 26262-6]:
   - Comparison of all clauses between [EN 50128] and [NF EN 62304]
   - Comparison of all clauses between [ISO 26262-6] and [NF EN 62304]

2. Opening of initial scope [NF EN 62304] with [IEC 60880]:
   - Identification of concepts not covered by the two standards (eg Security)
   - Comparison of coverage of these notions with [NF EN 62304]

In the same way as phase 2, a synthesis of these improvements is presented in chapter 4.2. Detail of these improvements has been grouped as sheets included in chapters 5.3, 5.4 and 5.5.

4.SUMMARY OF THE STUDY

Analysis led to the formulation of recommendations and improvements which is the subject of the present synthesis. For each of these recommendations / improvements, a detailed analytical study was prepared and is provided in the next chapter.

4.1. SUMMARY OF RECOMMANDATIONS

Recommendations in this report were carried out on the basis of analyzed incidents that are not at the origin of a proposed normative improvement. They provide guidance to medical device software manufacturers for the implementation of some requirements of [NF EN 62304] and [ISO 14971].

Summaries of recommendations are:

- Database - [NF EN 62304]:
  When software has a database, the specification of software requirements shall identify principles on data (sustainability, integrity, uniqueness and confidentiality).
  Use of a recognized DBMS and good practice rules is recommended.

- Specification / architectural design verification - [NF EN 62304]:
  It is recommended to perform and to formalize proofreading of produced documents, following several proofreading axes (completeness, relevance, consistency, traceability...).

- Impact analysis of a modification - [NF EN 62304]:
  Any modification of software shall be subject to an impact analysis to avoid defects induced by this change. It is recommended to treat the following points: Impact on existing MMR, addition of new MMR, impacted software items (EL), modification testing and non-regression testing.

- Management and monitoring of risks - [NF EN 62304] / [ISO 14971]:
  Risk analyzis have to be carried out at MD level and at software level (FMEA of the software).
  Implementation of a follow-up document of Risk Control Measures (MMR) is recommended to decide on the implementation and testing of these MMR.
  Use of the application guide of ISO 14971 [TR 80002-1] is recommended.

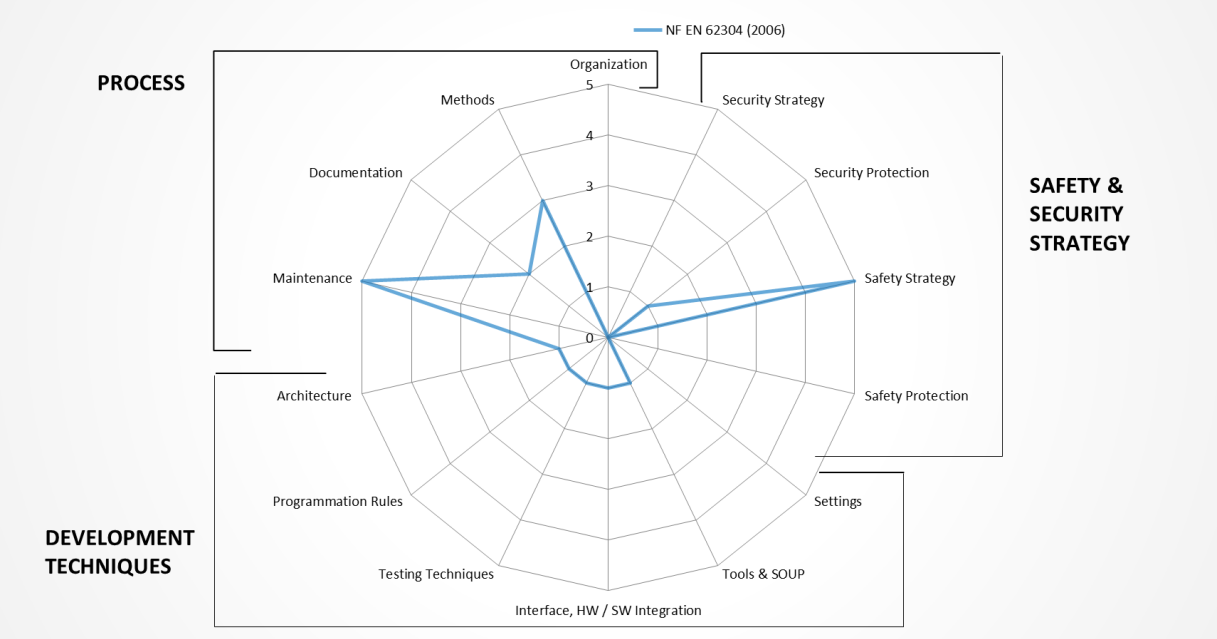
4.2. SUMMARY OF IMPROVEMENTS

Improvements proposed in this report are intended to cover the widest possible perimeter in terms of software typology (real-time constraint or not). Development activities associated with these differents typologies can be adapted when developing an evolution of the current standard.

For example, for software with no real-time constraints, activities concerning detailed design and unit testing may not be required depending on the risks identified through risk management activities defined in [ISO 14971].

4.2.1. [NF EN 62304] AND [ISO 14971] IMPROVEMENT

It should be reminded that positioning of [NF EN 62304] with regard to the state of the art, led to the finding below:



Suggested improvements allow to fill lacks identified in [NF EN 62304].

4.2.1.1. Process Category - [NF EN 62304]

For process category, these improvements focus on the following aspects:

- Organizational structure:
  Organizational structure shall be defined according to software safety class by specifying role and responsibilities of various stakeholders and independence between teams. Skill of stakeholders shall be demonstrated (technical knowledge, experience and suitable training).

- Assessment/Certification:
  An assessment process shall be added to standard. Assessor / certifier shall carry out a Software Assessment Plan (description of the assessment strategy) and Software Assessment Report acting on the compliance with the standard, on MMR and on identified anomalies.

- Global Traceability:
  Traceability between different activities (including MMR) shall be implemented, through a downward / upward traceability (from software requirements specification to code), and through a horizontal traceability to connect software tests with development documents.

- Deliverables:
  A table identifying documents to be produced by development activity according to software safety class shall be added in the standard. For each activity described in the standard, entry points and exit points has to be identified.

- Performance and environmental constraints:
  Sizing and performance constraints, physical characteristics, IT environment in which software works, shall be defined and controlled during installation.

- Installation:
  Specific tests shall be carried out and formalized to ensure that software is installed properly taking into account the runtime environment, required performance and demonstrating fulfilment of features.

- Detailed design:
  Detailed design shall describe Software Units (UL) and their interfaces (definition of input / output, domain of definitions, boundary values, algorithms, interfaces of call).

4.2.1.2.Technique of Development Category- [NF EN 62304]

For Technique of Development category, these improvements focus on the following aspects:

- General Techniques of Development:
  Tables shall be integrated into the standard in order to detail techniques to be implemented to carry out activities of development process. Techniques to apply shall depend on software safety class.

- Architectural constraints:
  A detailed description of safety architecture and fault tolerance strategy shall be implemented. To ensure robustness of software architecture, several approaches are possible (redundancy, segregation, defensive programming, limited interfaces, detection mechanisms and error handling).

- Software Unit (UL) implementation and verification constraints:
  Constraints shall be listed in the standard in a non-exhaustive way for programming languages that can be used and for programming rules.
  To ensure consistency of the coding activity, a check shall be established through a static code analysis (verification of programming rules) and a verification of consistency between detailed design and code.

- General techniques of test:
  Constraints for types of test, test coverage, techniques of test and testing environment, shall be defined. Furthermore, section 5.7 of [NF EN 62304] for Software Testing (Software Validation Tests) shall be applicable for software safety Class A.

- Unit Testing:
  For software safety Class C, activity of Unit Tests shall be added explicitly to requirements of the standard. This activity makes able to demonstrate that detailed design has been well implemented in the code, and that code does not contain non-desired functionality.

- Interfaces and Hardware / Software Integration:
  Specify elements of external interfaces description in consistency with safety constraints in its material environment and in interface with other software. For each software item, specify internal interfaces description. Integration tests shall be implemented to cover external and internal interfaces.

- Tools qualification:
  A specific section on tools qualification shall be added in the standard. Tools shall be classified according to the malfunctioning impact. Depending on this impact, tasks shall be carried out to qualify tool (realization of a User Manual, tool validation…). Demonstration of a successful prior use of tool in similar environments is also possible to qualify a tool.

- SOUP requirements grouping:
  A dedicated chapter with all requirements applying to SOUP shall be created.
  Furthermore, notion of pre-existing software is more appropriate than concept of SOUP. Its definition is « Software developed prior to the application currently in question, including COTS (commercial off-the shelf) and open source software ».

- Software configured by application data / Settings:
  A chapter dedicated to generic software configured by application data (parameters), as well as their verification and validation, shall be added to the standard. Software configured by application data shall have the same level of confidence that various possible sets of data.

4.2.1.3.Safety & Security category - [NF EN 62304] and [ISO 14971]

For Safety & Security category, these improvements include the following aspects:

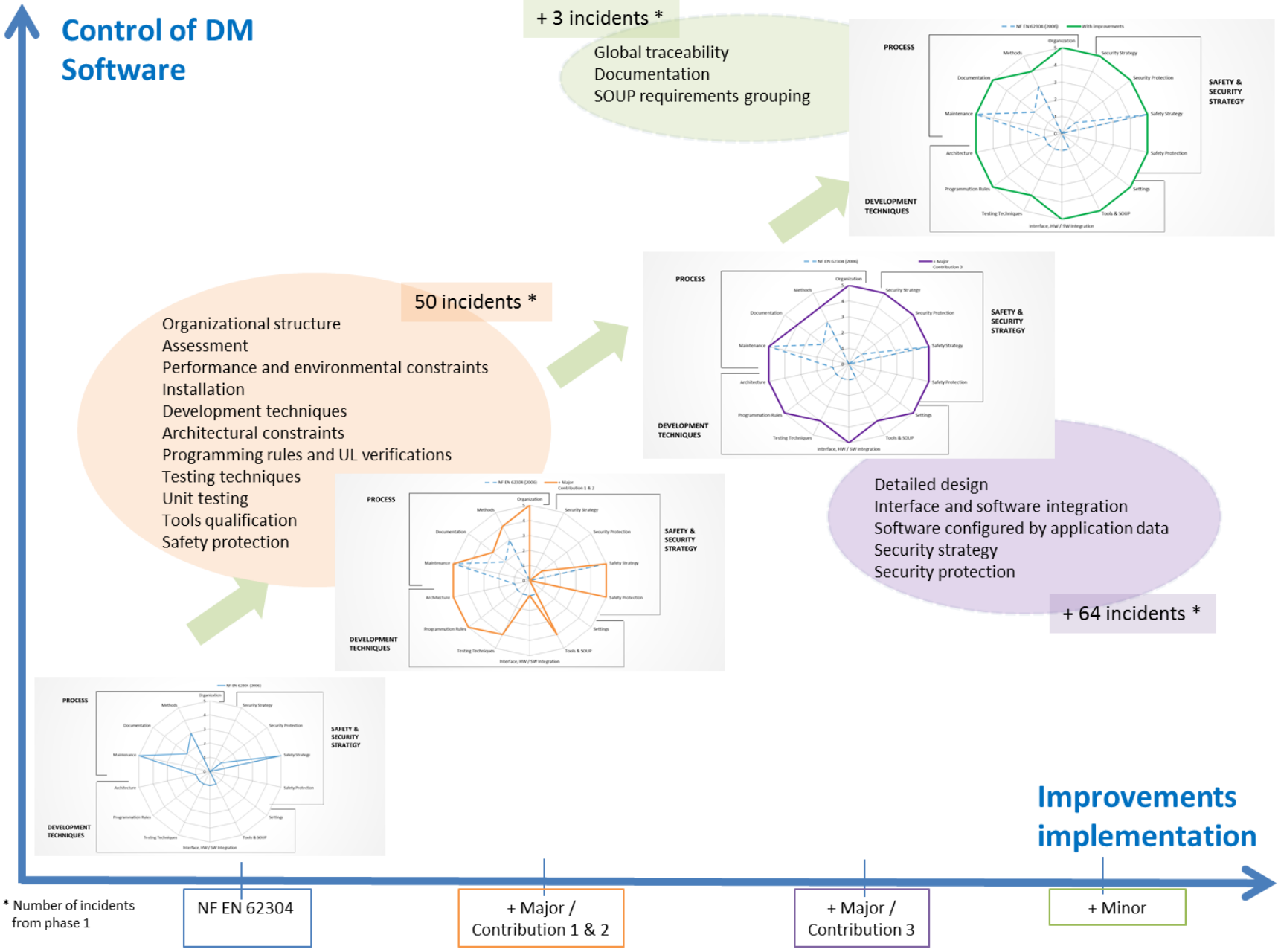
- Self-monitoring:
  Software shall perform self-monitoring to check the equipment during operation at specified time intervals including behavior of the software.
  In case of detection of a failure, software shall give appropriate responses in proper time (set to fallback position, degraded operation...).

- Guidelines for Security Strategy:
  Guidelines for Security Strategy (security processes) shall be addressed in [NF EN 62304] but a specific reference source treating this subject is needed. A threat prevention plan and a software security report shall be performed. In addition, applicable requirements throughout software lifecycle shall be integrated into activities (countermeasures requirements, authentication requirements, security vulnerabilities detection...).

- Security Protection:
  Design and development constraints on software security protections shall be defined (encryption, integrity monitoring, source authentication, no hidden paths). In addition, software development environment (workstation) shall be physically or digitally secured (partitioning, firewall, antivirus, data encryption...).

- Risk management and treatment of the issue of Security
  Risk management related to security shall be included in [ISO 14971].
  Treatment of Security issue has to be managed in a specific standard.

Suggested improvements can be gradually implemented according to the level of importance and contribution for manufacturer. Following diagram allows to visualize these improvements contribution for mastering of medical device software.


Control of DM Software
+ 3 incidents *
Global traceability
Documentation
SOUP requirements grouping
50 incidents *
Organizational structure
Assessment
Performance and environmental constraints
Installation
Development techniques
Architectural constraints
Programming rules and UL verifications
Testing techniques
Unit testing
Tools qualification
Safety protection
Detailed design
Interface and software integration
Software configured by application data
Security strategy
Security protection
+ 64 incidents *
PROCESS
SAFETY & SECURITY STRATEGY
DEVELOPMENT TECHNIQUES
Improvements implementation
* Number of incidents from phase 1
NF EN 62304
+ Major / Contribution 1 & 2
+ Major / Contribution 3
+ Minor

4.2.2. IMPROVEMENT OF USABILITY ENGINEERING [NF EN 62366]

Suggested improvements include the following aspects:

- User interface:
  User interface requirements shall be introduced in the normative part (§5.7) based on information provided in chapters D.2.6 (examples of user interface requirements), D.4.6.3 (specifications that may be included in a user interface), and in table G.7 (design details regarding alarm signals).

- Instructions for use:
  Provision of software user manual shall be required. In addition, requirements on formalization of a user manual shall be added to Chapter 6 (settings constraints, scope of use, limitations and restrictions of use).

- Users training:
  Chapter 7 (training and training materials) of the standard shall be completed in order to add constraints on training content, on verification of training materials and on evidence of training (eg : proof that medical staff has been trained and is able to use MD).

4.2.3. IMPROVEMENT VERSUS INNOVATION

For several years, the healthcare industry knows a real technological revolution. This revolution takes names such as:

- Digital Hospital
- Telemedicine / telesurgery
- Connected objects
- New technologies (nano-implants, Multi-heart, virtualization...)

In addition to economic or ethical issues, regulatory and normative issues arise for assessment of these medical devices applications.

Current standards must not impede technological progress but must try to anticipate consideration of new risks.
This aspect is relatively well taken into account in [ISO 14971] related to risk management. Indeed, this standard defines a generic approach applicable to any technological change.

Different kind of impacts on MD software related to the development of these new technologies can be identified:

- Impact on functionalities and / or response time (Safety and Security) of MD software related to use of multi-application software
- Impact on control and / or sustainability (Safety constraints) of MD software in connection with increased use of SOUP / COTS
- Impact on functionalities and data / privacy (Safety and Security constraints) related to a vulnerability

4.2.3.1. Using multi-application software

Development of multi-application software may in particular include use of technologies based on virtualisation or use of multi-heart systems.
These technologies are an opportunity for a greater consideration of Safety and Security constraints.
These solutions enable on the same platform implementation of applications with various criticality levels while ensuring their cohabitation. In addition, these technologies enable to limit exactly certification costs depending on the criticality level of each application.

Nevertheless, according to possibilities offered by these technologies, some limits will need to be considered:

- Loss of determinism
- Risks related to shared memory
- Execution time and data shared between tasks

Furthermore, use of these technologies is not yet accepted in all the fields. Indeed, several questions arise about ability to evaluate by a security point of view all behaviors of these systems.

Some requirements found in [NF EN 62304], and the various tracks for improvement suggested in this report can already be applied. These points cover in particular the following aspects:

- Modular software
- Architecture: Partition / separated shared memory (see §5.3.2.2)
- Settings (see §5.3.2.7)
- Limited Interfaces / integration (see §5.3.2.5)
- Specific programming rules for this type of application (see §5.3.2.3)

4.2.3.2. Using SOUP/COTS

In the medical field, use of SOUP / COTS is increasingly widespread.
Chapter 5.3.2.6 identifies a proposal of improvement for this purpose.

Moreover, besides proposal of improvement, initial choice of SOUP is crucial and shall take into account safety constraints where the SOUP will be integrated.

Indeed, risks management report shall ensure that SOUP used in software are appropriate for the concerned MD and that they comply with safety functional requirements of MD.

4.2.3.3. Integrating notions of Security

In the medical field, safety need of MD software is increasingly important because of widespread connectivity (wired or non-wired) of devices (Internet, local networks...), for needs like software updating, data feedback, interconnection with information systems, or remote control of device.

As part of software with real-time constraints, problem of security becomes even more important from the moment where they control some implanted devices guaranteeing therapy or maintenance of life of patients (immediate risk). Connectivity of these devices introduce real risks for security of software that control them.

More details about security concept are available in chapter 5.3.3.2.

Thus, it is appropriate to take action, not only in software development, but also throughout its lifecycle to ensure a minimum security level for software.

These measures are found in Chapters 5.3.3.2.2 and 5.3.3.2.3 in a form of proposals for improvement of [NF EN 62304] including consideration of security aspects in the lifecycle of medical devices software.

5.DETAIL OF RECOMMANDATIONS AND IMPROVEMENT AXES

5.1. ANALYSE FRAMEWORK

Recommendations and improvement axes are presented according to the following detailed sheet form:

Type	**Recommendation / Supplement / Creation**	Level of importance	**Minor / Major**
		Contribution	1 to 3
[Reference of the standard] - §			
Finding			
Recommendation or Improvement proposed			
Source	[EN 50128] ☐	[ISO 26262] ☐	[IEC 60880] ☐
	Incidents ☐	Other ☐	
Justification			

- **Type**
  Type concerning proposal of improvement for the standard, three types are defined:
  Recommendation: It is intended for MD manufacturers to implement paragraph of the standard cited in reference.
  Supplement: Concept is mentioned in the considered standard but more information is to be added.
  Creation: Concept is not mentioned in the considered standard.

- **Level of importance**
  Level of importance attached to recommendation / improvement proposal. Two levels are defined:
  Minor: recommendation / improvement is not seen with a high level of importance in the medical context.
  Major: recommendation / improvement is seen with a high level of importance in the medical context.

- **Contribution (Effort / human load) - *Only for improvement axes***
  Contribution from MD manufacturers to implement proposed improvement. Three levels of contribution are defined:

1: It is considered a low contribution to this improvement implementation because:
- Activity is already integrated in the majority of MD manufacturers,
- Low load to carry out activities.

2: It is considered a medium contribution to this improvement implementation because:
- Activity is already integrated in the majority of DM manufacturers,
- Heavy load to carry out activities.

OR
- Activity is low integrated for DM manufacturers,
- Medium load to carry out activities.

3: It is considered a strong contribution to the improvement implementation because:
- Activity is low integrated for DM manufacturers,
- Definition of process / methodology to implement,
- Heavy load to carry out activities.

- **Observation**
  For recommandations: observation concerning non-compliance with paragraph of the considered standard.
  For improvement axes: Finding concerning non-completeness, absence of the requirement, or a lack of concept in the standard.

- **Proposed recommendation / Improvement**
  Recommendation for considered standard implementation based on SERMA INGENIERIE experience feedback.
  Improvement proposal of considered standard based on [EN 50128], [ISO 26262] and [IEC 60880] selected standards, and on SERMA INGENIERIE experience feedback.

- **Source -** ***for recommendations, only the field "Incident" is completed***
  Identification of sources used to carry out improvement proposal.
  When standards are cited, chapter (or related chapters) is identified.
  For incidents, number of incidents related to proposed improvement considered as primary potential cause is indicated in brackets, if applicable.
  If another source than selected standards and incidents has been used, it will be quoted in the part called « Others ».

- **Justification**
  For recommendations: Justification explaining level of importance and selected incidents.
  For improvement axes: Justification explaining value of this improvement, but also various levels of importance and levels of the choosen contribution. This justification can be based on sources (Selected standards, incidents...)

5.2. RECOMMENDATIONS FOR MANUFACTURERS

5.2.1. DATABASE

Type	**Recommendation**	Level of importance	**Major**
[NF EN 62304] § 5.2.2 g) : Data definition and database requirements			
Finding	MD often need to manage a large amount of information: patient data, medical data, technical data... Some of them are even dedicated to this. They manage data to (patient data) or from (activity monitoring) other medical devices. For this, the MD is based on a database and DBMS. [NF EN 62304] - §5.2.2 g) requires manufacturers to establish database requirements. Some recommendations can be made on this subject.		
Recommendation	There are basics ensuring proper design of a database. Requirements of the software can bring up any or all of these concepts, which include: - Data continuity: Data must not be lost. - Data integrity: Values must be within permissible limits, in a correct and consistent format with what already exists in the database. - Data uniqueness: Data must not be redundant. - Data privacy: No unauthorized person shall have access to confidential data. Rules of best practice and use of a well-known DBMS are to be defined in order to implement concepts above. This concerns: - Data type specification, data structure… - Definition of consistency rules (consistent dates, no redundant information...) - Definition of a command language to manipulate database (query processing).		
Source	Incidents	2	
Justification	Two incidents analyzed during Phase 1 are due to mismanagement of database. The one results from a redundancy data in the database, the other from an absence of update of the database after a user action (such databases are internal to medical device). Level of importance of this recommendation is important regarding criticality and confidentiality of data managed by medical devices.		

5.2.2. SPECIFICATION / ARCHITECTURAL DESIGN VERIFICATION

Type	**Recommendation**	Level of importance	**Major**
[NF EN 62304] § 5.2.6 : Verify software requirements § 5.3.6 : Verify software architecture			
Finding	Requirements of §5.2.6 and §5.3.6 of [NF EN 62304] request to check activities of: - Defining software requirements (specification), - Architectural design. Several recommendations can be made on these requirements to guide MD software manufacturers.		
Recommendation	Related document shall be review to validate their adequacy towards input documents of each activity. Several review axes are possible, both on substance and form of documents. Verifications concern: - Completeness requirements; - Consistency (non-contradiction) requirements; - Requirements relevance; - Requirements traceability with upstream requirements; - Testability requirements; - Maintainability of the document; - Readability of the document. This reviewing has to be formalized in a document (Example: proofreading sheet) thus ensuring remarks follow-up. Remarks processing is the following: - Remarks made by reviewers; - Remarks taken into account or justified by the author of the document (according to the acceptance or rejection of the remarks); - Verification of the consideration and rationale of the author of the document for the remarks closure by reviewers. This treatment shall be repeated until remarks are closed or if an agreement has been found to treat remarks in a later version.		
Source	Incidents	7	
Justification	Seven incidents analyzed in Phase 1 are due to specification or architectural design that could have been avoided with proper verification: missing requirements, specific cases not taken into account … Level of importance of this recommendation is major. A proper verification of specification and architecture is essential because any mistake at this level can spread throughout software development cycle and can not be detected by tests.		

5.2.3. IMPACT ANALYSIS OF A MODIFICATION

Type	**Recommendation**	Level of importance	**Major**
[NF EN 62304] § 5.7.3 : Retest after changes § 7.4.2 : Analyse impact of software changes on existing risk control measures § 9.7 : Verify software problem resolution			
Finding	Modification of software may have an impact on parts of software that are not directly affected by change. Therefore, it is essential to measure this impact, to evaluate it from a perspective of « Risk », and to perform appropriate tests to validate modified software. [NF EN 62304] makes several requirements in that direction: - §5.7.3 talks about changes that occur during software testing phase. Manufacturer shall perform « appropriate tests » in order to demonstrate that « unexpected side effects » have not been introduced. - In §7.4.2, manufacturer shall determine whether modification of software can « interfere » with existing risk control measures. - §9.7 requires manufacturer to verify whether « Additional problems » have not been introduced due to implementation of a change. Furthermore, §5.6.6 asked to perform during software integration phase, an « appropriate regression test » to demonstrate that defects have been introduced in software previously integrated.		
Recommendation	Regardless of software lifecycle phase (development, operation, maintenance), any modification of software is subject to an impact analysis which shall enable: - To ensure that additional potential causes are not introduced. - To identify additional risks control measures if necessary. - To ensure that existing risks control measures are always processed. - To identify all Software Items (EL) potentially concerned by this change, especially « critical » EL (i.e., which can contribute to a hazardous situation). - To identify tests that shall be « replayed » (non-regression). - To achieve further specific tests related to change. Impact analysis is facilitated by correct architectural design and detailed design which enable to know interactions between Software Items and interactions between Software Units.		
Source	Incident	1	
Justification	An incident analyzed in Phase 1 is due to a fault introduced during the correction of another fault. Impact of change related to first fault has not been		

	properly determined. Level of importance of this recommendation is major considering criticality of medical devices software and increasing complexity of software which introduces edge effects during modifications, sometimes difficult to identify.

5.2.4. RISK MANAGEMENT AND MONITORING

Type	**Recommendation**	Level of importance	**Major**
[NF EN 62304] § 7 : Software risk management process [ISO 14971]			
Finding	Some anomalies in connection with requirements specification or architectural design may be result of a risk analysis that failed: - To identify correctly hazardous situations. - Assess associated risks. - Implement Risk Control Measures (MMR). - Check implementation of MMR. Recommendations may be made to §7 of [NF EN 62304] and [ISO 14971].		
Recommendation	Risk analysis realisation: Several dysfunctional analysis methods exist to identify risks associated with MD and their software. Some of these methods are presented in Appendix G of [ISO 14971] and are performed by the manufacturer. It concerns for example: - PRA: Manufacturer, at startup of development process, can perform a Preliminary Hazard Analysis (PHA) to identify hazards, hazardous situations, and events which can cause damages (for more detail see G.2 of [ISO 14971]) - FMECA: Following the division of MD into functional blocks, manufacturer may perform a Failure Modes, Effects and Criticality Analysis (FMECA). This analysis, based on the failure of functional blocks, enables to identify achievement or not of non-hazardous situations and to deduce from them appropriate MMR to secure various blocks of DM. We recommend application of FMECA methodology to software. This analysis enables to ensure: - Hazardous situations will not be achieved in case of software failure. - Adequacy of software protections. In this analysis, if hazardous situations are reached and protections are considered inadequate, this analysis helps identify new protections of Software in the form of MMR.		

	For software (identified in FMECA), analysis can be made at functionalities level (specification of Software), Software Items (Architectural Design), or at a lower level. For information, this technique of software FMECA is especially used in automotive and in railways fields under the name FMEA of the software. <u>Monitoring of Risk Control Measures:</u> For treatment of MMR, establishment by manufacturer of a monitoring document related to hazardous situations MMR is recommended. This document is initiated following risk analysis which has been realised upstream of software development. It is updated throughout software development cycle and grows rich in every activity. So, MMR are well monitored and traced, from specification to validation tests. Furthermore, it is recommended to use application guide of ISO 14971 [TR 80002-1].	
Source	Incident	1
Justification	An incident is due to poor identification of software items that can contribute to a hazardous situation. Level of importance of this recommendation is major. It is essential in the medical field to know how to evaluate risks inherent in software, to implement and to verify Risk Management Measures.	

5.2.5. SYNTHESIS OF RECOMMANDATIONS

Table below presents a summary of these recommendations:

Recommendation	Significance level	Incidents number (1)
[NF EN 62304]		
Data Base	Major	2
Specification / architectural design verification	Major	7
Impact analysis of one modification	Major	1
[NF EN 62304] and [ISO 14971]		
Risks management and tracking	Major	1

(1) : « Number of incidents » corresponds to incidents (identified in the first phase of this study) related to recommendations.

5.3. IMPROVEMENT AXES FOR [NF EN 62304]

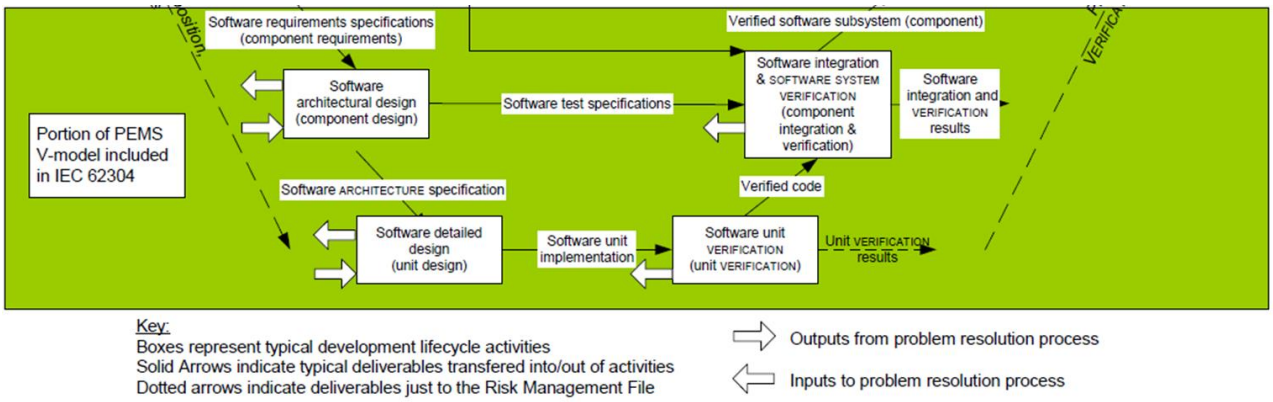
5.3.1. IMPROVEMENT AXES RELATED TO PROCESSES

5.3.1.1. Introduction to Lifecycle Process

[NF EN 62304] identifies several processes in Software lifecycle:

- Software Development Process,
- Software Maintenance Process,
- Risk management process related to Software,
- Configuration of Software Management Process,
- Software Problem Resolution Process.

For better understanding proposed improvements, it is appropriate to introduce concept of activities (also called development phases) for the part Software Development Process (§5 of the standard) by integrating the part related to Software in Figure C.2 of [NF EN 62304]. This figure is present in an informative part of the standard:



In order to synthesize this figure, software development activities identified in [NF EN 62304] - §5, besides planning phases of software, are:

- Software requirements analysis (Software Requirements Specification)
- Software architectural design (Software Items)
- Software detailed design (Software Unit)
- Software unit implementation (Coding)
- Software unit verification
- Software integration and software integration tests
- Software system testing (Software Validation Tests)

[NF EN 62304] references [ISO 14971] on the Risk Management Process (for more details, see §5.3.3).

In introduction of [NF EN 62304], it is stated that maintenance process is as important as development process (for details see §5.3.1.5).

Process of software configuration management and software troubleshooting are subject of a specific chapter in [NF EN 62304]. They have been found sufficient in this study.

5.3.1.2.Details of the Lifecycle

In introduction of [NF EN 62304], no specific lifecycle model is imposed. However, standard states that software development lifecycle shall be defined (§5.1.1 of the standard) and specifies that different development strategies (waterfall, incremental...) can be used (Appendix B.1.1 of standard).
This approach seems to us consistent and tailored to various types of software that MD software manufacturers can meet.
This information was deemed sufficient and is not indicated in the improvement proposals.

5.3.1.3.Organization

5.3.1.3.1.Organizational structure

Type	**Creation**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] (creation of a chapter in §4 : General requirements)			
Finding	As stated in introduction to [NF EN 62304], no requirement concerning organizational structure is addressed in the standard. Concepts of organization, roles, responsibilities, skills and independence of teams or stakeholders are not defined.		
Proposed improvement	Organizational structure shall be set according to software safety class, specifying roles and responsibilities of various stakeholders. For example, based on [EN 50128], stakeholders may be: - Project manager, - Requirements Managers / Designer / Developer, - Responsible for Integration / Responsible for Validation (Tests), - Verifier, - Responsible for activities related to Risk… Appendix B of [EN 50128] describes responsibilities and main skills of stakeholders. Example of responsibilities of a designer: shall transform specified requirements in acceptable software solutions. Independence between teams shall be set depending on the class of software (diagram of organizational structure based on classes). For software safety Class C, independence between development team and test		

	team is required. As well as for team in charge of risks management. Skills of stakeholders are to be defined in the standard. Manufacturer shall provide documented proof of skill of staff (technical knowledge, experience, and appropriate training). It is necessary for each stakeholder to demonstrate a continuous preservation and a development of skills. This part is discussed in §6.2 of [ISO 13485].		
Source	[EN 50128] ☒ § 5.1, 5.2, Appendix B	[ISO 26262] ☒ Part 2	[IEC 60880] ☐
	Incidents ☐	Other ☐	
Justification	Definition of the organization, roles and responsibilities, aims to ensure that all staff responsible of software is organized, able to assume its responsibilities and competencies to fulfill these responsibilities. Dealing with this last point, manufacturer shall demonstrate ability to perform these tasks correctly, efficiently and consistently at a high quality level (Source [EN 50128]). Thus, level of importance is major. Contribution for manufacturer is medium, considering that independence into organization requires multiple stakeholders.		

5.3.1.3.2.Assessment / Certification

Type	**Creation**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] (creation of a dedicated chapter)			
Finding	No requirement is applicable to assessor / certifier for software lifecycle processes in [NF EN 62304].		
Proposed improvement	An assessment process shall be added to the standard. The assessor / certifier shall conduct Software Evaluation Plan which defines: - Strategy and assessment methodology (audit of activities and stakeholders, documentary assessment, tests assessment...), - Documents to assess, - Requirements related to the content of Software Assesment Report. Assessor / certifier shall also perform a Software Assessment Report, which, based on the risk management report, shall rule on: - Compliance with [NF EN 62304] according to software safety class, - Risk control measures (MMR),		

<table>
<tr><td></td><td colspan="3">- Product documentation,
- Identified anomalies.
It shall especially be mentioned assessed version, restrictions and constraints of use. This report is only valid for a specific version.

Activity shall be conducted by an independent entity and is based on software safety class.

Assessor / certifier shall be involved at an early stage of the project.</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] §6.4 ☒</td><td>[ISO 26262] ☐</td><td>[IEC 60880] ☐</td></tr>
<tr><td>Incidents ☐</td><td colspan="2">Other ☐</td></tr>
<tr><td>Justification</td><td colspan="3">This activity aims to assess that development cycle processes and their outputs are such that software complies with software safety class and is suitable for its intended application.
Description of assessment process will give a guideline to assessors / certifiers and will indicate to manufacturer points that will be evaluated.
Contribution for manufacturer is medium because it imposes constraints to assessors / certifiers and may generate additional demonstrations to set up by manufacturer; but it will give a framework for activities to be undertaken by assessor / certifier.</td></tr>
</table>

5.3.1.4.Methods

5.3.1.4.1.Global traceability

Type	**Complement**	Level of importance	**Major**
		Contribution	1
[NF EN 62304] § 5.1.1 : Software development plan			
Finding	Only traceability between system requirements, software requirements, software testing (Validation Test) and MMR implemented in the software is being treated.		
Proposed improvement	Traceability between activities (including MMR) shall be implemented through: - A downward and upward traceability from the Software Requirements Specification to the code: ✓ Requirements Specification <-> Architectural Design ✓ Architectural Design <-> Detailed Design ✓ Detailed design <-> Code - A horizontal traceability in order to connect software testing with development documents: ✓ Detailed design <-> UL Verification ✓ Architectural Design <-> Software Integration ✓ Requirements Specification <-> Software Testing Traceabilities shall be checked.		
Source	[EN 50128] ☒ §5.3.2.7, 6.5.4.14, 6.5.4.15	[ISO 26262-6] ☒ §7.4.2, 8.4.5	[IEC 60880] ☐
	Incidents ☒ (3)	Other ☐	
Justification	Traceability aims to demonstrate end-to-end monitoring of all software requirements (including requirements related to MMR), and therefore control of development and testing activities. Traceability also has benefit of easier maintainability. Contribution of manufacturer is considered low because it does not require a particular investment. Moreover, traceability is made by the majority of manufacturers. Furthermore, incident analysis has identified three cases in conjunction with a lack of traceability.		

5.3.1.5.Documentation

5.3.1.5.1.Deliverables

Type	Creation	Level of importance	Minor
		Contribution	1
[NF EN 62304] (creation of an Appendix)			
Finding	[NF EN 62304] does not identify a clear and concise manner to provide documents as part as a software development. Moreover, documentary inputs and deliverables to be produced for each development activity (eg: Architectural design) are not identified in the standard.		
Proposed improvement	A table identifying documents to be produced by development activities based on software safety class is to be added. These tables are given in [EN 50128] and [ISO 26262-6]. Moreover, for each activity in the main section of the standard, identify entry points and exit points in the standard.		
Source	[EN 50128] ☒ Appendix A.1	[ISO 26262-6] ☒ Appendix A	[IEC 60880] ☐
	Incidents ☐	Other ☐	
Justification	Information concerning documents having to be produced for each activity depending on software safety class allows to clarify documents having to be produced by manufacturer. Contribution for manufacturer is low because it does not add extra load.		

5.3.1.5.2.Performance and environmental constraints

<table>
<tr><td rowspan="2">Type</td><td rowspan="2">Complement</td><td>Level of importance</td><td>Major</td></tr>
<tr><td>Contribution</td><td>2</td></tr>
<tr><td colspan="4">[NF EN 62304]
§5.2.2 – a) : Functional and capability requirements</td></tr>
<tr><td>Finding</td><td colspan="3">Requirement §5.2.2 -a) of [NF EN 62304], indicates that requirements in terms of functionality and capacity shall be defined.
Note 1 identifies information about this subject, which are merely « examples ».</td></tr>
<tr><td>Proposed improvement</td><td colspan="3">Note 1 should be included in requirement of §5.2.2 - a) to clearly identify constraints having to be inform by the manufacturer.
This concerns:
- Constraints of dimensioning and performance: Response time, synchronization…
- Physical characteristics: Language, operating system…
- Computer environment where software shall work: memory space constraints, network infrastructure, CPU, hard disk size…

These constraints shall be covered by checks / tests by the manufacturer and for some, checked during installation (see related improvements to facility).
Testing techniques are listed in [EN 50128] (Table A.18) and [ISO26262-6] (Table 13 and 16).</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] ☒
§6.2.4.7, §7.2.4.2, §7.2.4.19, Table A.18</td><td>[ISO 26262] ☒
§11, Table 13, 16</td><td>[IEC 60880] ☒
Appendix B2</td></tr>
<tr><td>Incidents ☒ (4)</td><td colspan="2">Other ☐</td></tr>
<tr><td>Justification</td><td colspan="3">Definition of these constraints role is to prevent occurrence of incidents that may have as primary cause:
- Failure to respect performances (memory, CPU, hard drive size...),
- Non-control of environment – not functional software due to an external cause of software (network overload, software interoperability problem, undersized processor...).
4 incidents occurred, induced by these causes.
Contribution for manufacturer is considered moderate because constraints have to be formalized and validated with the manufacturer and during the installation.</td></tr>
</table>

5.3.1.5.3.Installation

Type	**Complement**	Level of importance	**Major**
		Contribution	1
[NF EN 62304] § 5.2.2 – h) : Installation and acceptance requirements of the delivered medical device software at the operation and maintenance site or sites			
Finding	[NF EN 62304] contains few requirements for installation and verification. Only requirement in §5.2.2 - h) indicates that manufacturer shall include requirements of software installation and acceptance for MD delivered to sites of operation and maintenance.		
Proposed improvement	[NF EN 62304] shall integrate requirements to control on-site installation. Specific tests shall be performed to ensure software is installed properly taking into consideration runtime environment, required performance and demonstrating respect for features. These tests shall be formalized.		
Source	[EN 50128] ☐	[ISO 26262] ☒ Table 16	[IEC 60880] ☒ § 12
	Incidents ☒ (4)	Other ☐	
Justification	Formalization of installation activities will ensure proper installation considering functional and performance standpoint, and execution environment. 4 incidents are due to improper installation. Contribution for manufacturer is considered low because, apart from formalizing, these activities are already performed by most manufacturers.		

5.3.1.5.4.Detailed design

<table>
<tr><td rowspan="2">Type</td><td rowspan="2">Complement</td><td>Level of importance</td><td>Major</td></tr>
<tr><td>Contribution</td><td>3</td></tr>
<tr><td colspan="4">[NF EN 62304]
§ 5.4 : Software detailed design</td></tr>
<tr><td>Finding</td><td colspan="3">Requirements of §5.4 of [NF EN 62304] require development and documentation of each Software Unit (UL) with documentation of interfaces between UL.
However, no specific constraint on description of UL and of these interfaces is present in the normative part.
The §B.5.4 in informative Appendix B provides additional information.</td></tr>
<tr><td>Proposed improvement</td><td colspan="3">Additional information of §B.5.4 shall be included in §5.4 of the normative part, and especially the following part:
« The detailed design specifies algorithms, data representations, interfaces between different UNITS SOFTWARE and interfaces between UNITS SOFTWARE and data structures. »
To be complete, needed information to describe a UL are as follows:
- Definition of I/O data of UL (parameters, global variables, back...),
- Definition and description of areas of definitions of I/O (data equivalence classes),
- Definition of boundary value data and behavior for exceeding,
- Detailed description of algorithms,
- Call interface between UL (function call).

Furthermore, [EN 50128] incorporates fact that size and complexity of each component (UL) shall be balanced.</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] ☒
§7.4.4.1 à 7.4.4.5</td><td>[ISO 26262-6] ☒
§8.4.4</td><td>[IEC 60880] ☐</td></tr>
<tr><td>Incidents ☒ (12)</td><td colspan="2">Other ☐</td></tr>
<tr><td>Justification</td><td colspan="3">Implementation of these requirements will address the following part of the standard §B.5.4:
« Detailed design informs of needed details to build software. It should be completed enough in order to programmer does not have to take circumstantial design decisions. »

In addition, 12 incidents are due to the lack of description of detailed design, considered as potential primary cause.

Furthermore, detailed design is also the entry point for UL verification activity and a lack of information does not allow identifying bugs on particular execution paths.</td></tr>
</table>

	Contribution for manufacturer is strong, because an effort of UL description shall be done. After that, effort shall focalize on testing for UL verification.

5.3.1.6.Maintenance

No specific improvement is proposed.

5.3.2. IMPROVEMENT AXES RELATED TO DEVELOPMENT TECHNIQUES

[NF EN 62304] contains few constraints on development techniques concerning:

- Architectural constraints,
- Implementation constraints,
- Testing techniques,
- Interfaces and integration HW / SW constraints,
- Application data constraints (settings).

These constraints are subject of various proposals for improvements present in this chapter. They need to be adapted according to software safety class.

Nevertheless, [NF EN 62304] uses the concept of software item in architectural design activity and concept of Software Unit in detailed design activity.
These concepts help to realize a modular and structured software development. This is a good point in the development of critical software.

5.3.2.1. General Development Techniques

Type	**Creation**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] (creation of an Appendix)			
Finding	[NF EN 62304] do not contain any requirement about development techniques to achieve.		
Proposed improvement	In [EN 50128], tables (A1 to A23) of the Appendix A (normative) shows in detail techniques to be implemented in order to achieve development activities according to Software criticality level. For each technique, a link to an explanation (objective and description) is present. [ISO26262-6] also has the same approach in Tables 1 to 16. This type of table shall be integrated in [NF EN 62304] adapting their content to medical field. Techniques shall be applied according to software safety class by establishing several levels: - Mandatory; - Highly recommended; - Recommended; - No recommendation for or against its use; - Not recommended. Some Mandatory or Highly Recommended techniques, extracted from [EN 50128], and dedicated to the Software Architecture activity are given here (non		

	exhaustive list): - Defensive programming, - Fault detection & diagnosis, - Assertion programming, - Diversified programming, - Fully defined interface, - Structured methodology… Some techniques, extracted from [EN 50128], and dedicated to design and implementation activities of software (not exhaustive) are given here: - Structured methodology, - Modular approach, - Design and coding rules, - Structured programming, - Subset language…

Source	[EN 50128] ☒ Tables A.1 to A.23 (Appendix A)	[ISO 26262-6] ☒ Tables 1 to 16	[IEC 60880] ☐
	Incidents ☐	Other ☐	
Justification	Use of such type of tables would illustrate how to ensure compliance with requirements of the standard. In addition, a dedicated normative Appendix integrating these tables would be an asset for manufacturers to have clear guidelines for Software realisation. Thus, degree of importance is considered major. Contribution for manufacturer is medium because of application of the defined techniques.		

5.3.2.2. Architectural constraints

Type	**Complement**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] § 5.3 : Software architectural design			
Finding	Few architectural constraints are present in [NF EN 62304]. In §5.3.5, a constraint is present for a software safety Class C: The separations between essential software items shall be identified for risk control as well as the methodology for ensuring that separation is effective (Class		

	C - §5.3.5).		
Proposed improvement	For development of software safety Class C, architectural constraints shall be added. Many concepts from [ISO26262] are to be integrated into [NF EN 62304]. A detailed description of safety architecture and fault tolerance strategy (related to criticality, hardware, system architecture constraints, safety requirements...) has to be implemented ([ISO 26262-5] Appendix D). To ensure robustness of software architecture, several approaches are possible: - Redundancy. Implementation of this technique improves safety and/or availability of software. - Segregation, partitioning and cohabitation of software components with various criticality levels (§5 and §6 [ISO 26262-9] for an ASIL decomposition of a system, and criteria for coexistence of elements with various ASIL). - Diversified programming. - Limited interfaces between safety and non-safety software. - Limited interfaces between safety and non-safety software items. - Error detection mechanism: ✓ Tasks sequencing control ✓ Time control (eg cycle counter): [ISO 26262-6] - D.2.2 ✓ Plausibility/limited data control (ex: assertion, comparison): [ISO 26262-6] - D.2.4 ✓ Data integrity check (ex: null pointers, CRC, checksum): [ISO 26262-6] - D.2.4 ✓ Memory Check (RAM, MMU, MPU): [ISO 26262-6] - D.2.3 ✓ Self-tests (watchdog, channels comparison, HW,...) and communication protocol - Error handling mechanism: ✓ Functions putting the system in a safe state, and degraded modes / fallback position ✓ Functions for monitoring and diagnostic of detected errors: • Alarm management, • Historical management, • Buffers of errors.		
Source	[EN 50128] ☒ Table A.3, Appendix D	[ISO 26262] ☒ Part 5: Appendix D Part 6: §D.2.2, D.2.3, D.2.4 Part 9: §5, 6	[IEC 60880] ☒ Appendix B2, B3

	Incidents ☒ (5)	Other ☐
Justification	Architectural constraints on fault tolerance strategies and protections help to ensure security of software with real-time constraint (determinism fallback position, consistent output value...). This point is crucial for software execution in a safe condition. Furthermore, a lack of robustness in software architecture is considered as the primary cause of 5 incidents. Contribution for the manufacturer is medium due to: - Establishing of robust architectures and protections in software, - Description of architectural choices in the documentation.	

5.3.2.3. Implementation constraints and UL verifications

Type	**Complement**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] § 5.5 : Software unit implementation and verification			
Finding	Acceptance criteria of Software Units are described in §5.5 and are depending on software safety class. There is a lack of details for these criteria. A note in §5.5.3 gives examples: – Does software code correctly implement requirements, including MMR? – Is software code contradictory to interfaces described in the detailed design documents of software unit? – Does software code meet programming procedures and coding standards?		
Proposed improvement	This note shall be converted in requirement, because consistency of the code with requirements (including MMR), detailed design, and programming rules is necessary for MD development. Similarly, a requirement shall be created to indicate that code shall be established by following design and programming rules defined by the manufacturer. [EN 50128] and [ISO 26262-6] identify this constraint and list in non-exhaustive way of design and programming rules to follow, as well as programming languages that can be used depending on the Software Safety and Integrity Level (SSIL). For example, [EN 50128] prohibits the use of such unconditional connections. [ISO 26262-6] highlights in an example MISRA-C reference source for C language.		

	Moreover, in order to ensure that code complies with the defined programming rules, [EN 50128] and [ISO 26262-6] request a static code analysis (this can be achieved via a verification tool for the application of programming rules).		
Source	[EN 50128] ☒ §7.3.4.25, §7.3.4.26, Tables A.12, A.19	[ISO 26262-6] ☒ §8.4.4, 8.4.5	[IEC 60880] ☐
	Incidents ☒ (20)	Other ☐	
Justification	All programming languages and instructions of these languages can not be used in critical development because it can undermine the reliability, maintainability, availability and safety of the software. Poor implementation of Software Units is considered as potential primary cause for 20 incidents. Origins can be: - Non-compliance or non-definition of design and programming rules, - Non-fulfillment of static code analysis, - Non-fulfillment of activities ensuring consistency between code and requirements (including MMR) and / or detailed design. Thus, level of importance is considered major, with a contribution for manufacturer considered moderate.		

5.3.2.4.Testing Techniques

5.3.2.4.1.General testing techniques

Type	**Creation**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] (creation of an Appendix)			
Finding	In the normative part of [NF EN 62304], information on testing techniques are absent. In informative Appendix B, only §B.5.4 identifies « black box » and « white box » concepts. As this part is present in informative Appendix, type of improvement proposal is « Creation ».		
Proposed improvement	In addition to the levels of black box testing, white box testing... several constraints and concepts have to be added: - Constraints on the types of tests. Example: ✓ Nominal Tests, ✓ Robustness tests, ✓ Performance tests… - Constraints on the coverage test. Example:		

	✓ Functional coverage, ✓ Structural coverage… - Constraints regarding testing techniques. Example: ✓ Equivalence class and score entries, ✓ Boundary value tests… Constraints on the test environment also have to be added (Eg: Host or target tests - in its execution environment). These concepts can be used for all phases of testing (Software Units Verification, Integration Testing and Software Testing). [EN 50128] and [ISO26262] introduce these concepts in a detailed way. They are applicable depending on the level of Safety Integrity of the Software (SSIL). This can be adapted to the software safety class. Furthermore, section 5.7 of [NF EN 62304] for Software Testing (Software Validation Tests) shall be applicable for software safety Class A software.

Source	[EN 50128] ☐ §7.4.4.9, Table A.14, Appendix D	[ISO 26262] ☒ §9, 10, 11	[IEC 60880] ☐
	Incidents ☐	Other ☐	
Justification	Introduction of these concepts will allow manufacturers a better understanding of testing activities (Verification of Software Unit, Integration and Testing Software Testing). Importance level is considered major with a contribution for the manufacturer and considered moderate as this will impact the definition of testing activities and its application in testing activities.		

5.3.2.4.2.Unit testing

Type	**Complement**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] § 5.5.4 : Additional software unit acceptance criteria			
Finding	The additional acceptance criteria of Software Unit described in §5.5.4 suggests Unit Tests activity for software safety Class C but the fact that it is not mentioned explicitly provides significant blur.		

<table>
<tr><td>Proposed improvement</td><td colspan="3">For software safety class C, Unit Tests activity has to be added to requirements of [NF EN 62304] as it is in [EN 50128], [ISO26262-6] and [IEC 60880].

This activity is also requested in the table 3 of FDA standard « Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices » of May 2005, from the « Moderate » level (Minor injury to the patient or operator).

[EN 50128] and [ISO26262] introduce concepts of instructions, branches, and conditions coverage. They are applicable depending on the Software Safety and Integrity Level (SSIL). This can adapt to the software safety class.</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] ☒
§7.5, table A.21</td><td>[ISO 26262] ☒
§9</td><td>[IEC 60880] ☒
Appendix B.4.g</td></tr>
<tr><td>Incidents ☒ (4)</td><td colspan="2">Other ☐</td></tr>
<tr><td>Justification</td><td colspan="3">Unit testing activity role is to demonstrate through a dynamic analysis (Software execution) that Software Units as described in the detailed design have been implemented in the source code (test results) and that code does not contain non-desired functionality.
Indeed, this activity ensures that all instructions, branches, conditions are well executed (no dead code, and passing through all execution paths of Code).
This is useful to identify quickly origin of an incident in the operation.
Furthermore, lack of unit tests is potential primary cause of 4 incidents.
Level of importance is considered major.
Contribution for the manufacturer is considered moderate because it is an activity with a heavy load, but this activity is already performed by most of manufacturers.</td></tr>
</table>

5.3.2.5.Interface, Hardware / Software integration

Type	Complement	Level of importance	Major
		Contribution	3
[NF EN 62304] § 5.3.2 : Develop an architecture for the interfaces of software items			
Finding	Requirement §5.3.2 of [NF EN 62304] calls for development and documentation of internal interfaces (among Utilities Software) and external interfaces to software items (both hardware and software) in architectural design. However, no specific constraint on the definition of interfaces is present in the normative part. §B.5.3 in informative Appendix B provides additional information without indicating what criteria for defining interfaces are.		
Proposed improvement	External interface: In terms of architectural design or in a specific document as HW/SW interfaces specification (for reuse needs), specify description elements of external interfaces consistent with safety constraints: - In its material environment, - Interfacing with other software. Following information are to be provided: - Medium used to transfer data: Physical interfaces, protocols… - Content exchanged: Logic interface, message format (see section internal interface for describing interfaces), - Use of material resources: Memories, registers, timers, interrupts, input / output ports, time constraints (critical time). Internal interface: In terms of architectural design, and for each software item, specify description of internal interfaces elements: - Definition of EL I/O data, - Definition and description of I/O definition areas (data equivalence classes), - Definition of boundary value data and behavior for exceeding, - For input and output data at critical time: ✓ time constraints and requirements for proper operation, ✓ exception handling - Existence of synchronization mechanisms between functions (see previous point), Integration tests are set up to cover external and internal interfaces (§5.6.2 [NF EN 62304]).		

Source	[EN 50128] ☒ §7.3.4.19	[ISO 26262] ☒ Part 4: §7.4.6	[IEC 60880] ☐
	Incidents ☒ (47)	Other ☐	
Justification	[EN 50128] and [ISO 26262] address this issue in detail. Lack of description of interfaces is potential primary cause of 47 incidents. This explains the level of importance is major. Contribution for manufacturer is considered high due to an effort having to be realized for interfaces description, and then integration testing.		

5.3.2.6.Tools & SOUP

5.3.2.6.1.Tools qualification

Type	**Creation**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] (creation of a dedicated chapter)			
Finding	There is no requirement for qualification of tools used during activities (development, validation, configurable generic software...) in [NF EN 62304]. Only one mention indicates that tools shall be checked at §5.1.10.		
Proposed improvement	3 standards selected for the study contain a specific section concerning tools qualification. Tools are classified according to the impact of a malfunction. 3 classes of tools are defined: - Class 1: A malfunction has no impact on executable code (including data). Example: Text editor, design tool… - Class 2: Tool used for testing or verification. A malfunction might not detect a defect without directly creating an error into the executable software. Example: Testing tool, static analysis… - Class 3: Generates outputs that can contribute directly or indirectly to executable code (including data). Example: Compiler… Tasks to be carried out for tools qualification are also depending on Safety class of software. For class 1 tool, choice of tool shall be justified. For class 2 tool, in addition to tool class 1: - Tool shall have a user manual or a specification defining its features and constraints of use.		

<table>
<tr><td></td><td colspan="3">
- An evaluation of tool confidence level shall be performed (study of potential risks, preventive measures, validation of tool...) or certificate.
- Tool shall be managed in configuration.
- Each new tool version shall be justified.

For class 3 tool, in addition to class 1 and 2 tools:
- Proof of compliance with specification of tool shall be made.
  OR
- Proof of establishment of measure to guard against failures of tool shall be provided (eg redundant code generation...).
  OR
- Use a certified/ qualified compiler.

Proof of a prior successful use of tool in similar environments is possible to qualify tool.

A tool qualification report has to be made containing tasks or demonstrations mentioned above. Versions of tools shall be identified.
</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] ☒
§ 6.7</td><td>[ISO 26262] ☒
Part 8: §11</td><td>[IEC 60880] ☒
§ 14</td></tr>
<tr><td>Incidents ☐</td><td colspan="2">Other ☐</td></tr>
<tr><td>Justification</td><td colspan="3">
The aim is to provide evidence that potential failures of tools do not undermine security of data produced by all tools without being detected ([EN 50128] - § 6.7.1).
Level of importance is considered major.

Contribution for manufacturer is medium because tasks to be performed to qualify tools are important but reusable on many projects (in case of using qualified version of tool).
</td></tr>
</table>

5.3.2.6.2.SOUP requirements grouping

<table>
<tr><td rowspan="2">Type</td><td rowspan="2">Complement</td><td>Level of importance</td><td>Minor</td></tr>
<tr><td>Contribution</td><td>1</td></tr>
<tr><td colspan="4">[NF EN 62304] (creation of a chapter in §5 : Software development process)</td></tr>
<tr><td>Finding</td><td colspan="3">Requirements for SOUP (Software Of Unknown Provenance) are scattered throughout [NF EN 62304]. Over 12 chapters contain requirements related to SOUP. There are identification requirements of definition, integration, risk management activities, maintenance…</td></tr>
<tr><td>Proposed improvement</td><td colspan="3">A dedicated chapter with all SOUP applicable requirements shall be performed.

Furthermore, notion of SOUP used in medical field is simplistic and does not provide any guarantee because:
- it is not developed to be integrated in developed system,
- it does not meet implementation process of applicable standard,
- records of development processes are insufficient or unavailable.

[EN 50128] uses notion of pre-existing software and has a dedicated chapter (§7.3.4.7) defining pre-existing software usage restrictions.
Definition of pre-existing software in [EN 50128] is « Software developed prior to the application currently in question, including COTS (commercial off-the shelf) and open source software ».
Unlike SOUP, COTS:
- is defined by needs of market
- its adaptation to needs is demonstrated by a wide range of users
- has an accessible and comprehensive documentation

Specific chapters are also present in [ISO 26262] and IEC [60880].</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] ☒
§7.3.4.7</td><td>[ISO 26262] ☒
Part 8: §12</td><td>[IEC 60880] ☒
§15</td></tr>
<tr><td>Incidents ☐</td><td colspan="2">Other ☐</td></tr>
<tr><td>Justification</td><td colspan="3">This improvement aims to clarify the concept of SOUP and facilitate application of those requirements by the manufacturer.
Level of importance is considered minor and contribution for manufacturer is low.</td></tr>
</table>

5.3.2.7.Setup - Software configured by application data

Type	Creation	Level of importance	Major
		Contribution	3
[NF EN 62304] (creation of a dedicated chapter)			
Finding	[NF EN 62304] contains no requirement for software configured by application data (parameters) and their verification and validation.		
Proposed improvement	A chapter dedicated to generic software configured by application data shall be added to standard. [EN 50128] contains a specific chapter on this concept in which all development cycle phases are addressed. The chapter to add shall contain the following requirements: - Implementation of an application preparation plan defining: ✓ Process to implement for each specific application or for each class of specific applications, ✓ Techniques and measures to implement (Specification methods with tables...), ✓ Description of verification and validation activities (compatibility of application data with generic software). - Application's Specification and Design: ✓ Requirements related to installation conditions of application data, ✓ Requirements for application data, ✓ Location of application data, ✓ Precise description of application data (values, type, domain, description...). - Application's verification and validation via: ✓ Control List, ✓ Functional testing datasets (equivalence classes Tests and scores entries, the limit values...), ✓ Proof of correctness of data and their integration. Generic software shall: - Identify functions configurable with application data, - Be separated from application data, - Perform a check on integrity and consistency of application data, when possible.		

	Modification control procedures shall ensure that in case of change in generic software, software can be installed if it is compatible with original application data or if data has been updated. If an application preparation tool is used, it must be qualified (see qualification of tools sheet).		
Source	[EN 50128] ☒ § 8	[ISO 26262] ☒ Appendix C	[IEC 60880] ☒ §7.1.4, 8.2.3.3, 14.3.5
	Incidents ☒ (5)	Other ☐	
Justification	Many MD software are architected with software indicated as "generic" which is configured with specific data (parameters). Configured software shall have the same level of confidence whatever the various possible sets of data. Furthermore, poor management of application data is primary cause of 5 incidents. Thus, degree of importance is major. Contribution for manufacturer is strong because a process is to be defined and to apply for each specific application or for each class of specific applications.		

5.3.3. IMPROVEMENT AXES STRATEGIES RELATED TO SAFETY AND SECURITY

[NF EN 62304] references [ISO 14971] on Risk Management Process.
[ISO 14971] describes Risk Management Process in detail (see chapter "Area for improvement [ISO 14971]").

Regarding safety-related protections, [NF EN 62304] does not define architecture, safety techniques/rules, and fault-tolerance strategy to make robust and to secure the software. This point is subject of proposed improvements.

In addition, [NF EN 62304] identifies few constraints related to security strategy and protection. This point is also subject of proposed improvements.

5.3.3.1.Safety part

5.3.3.1.1.Safety strategy

No specific improvement is proposed.

5.3.3.1.2.Safety protection

Type	**Complement**	Level of importance	**Major**
		Contribution	2
[NF EN 62304] § 5.2.3 : Include risk control measures in software requirements			
Finding	Requirement §5.2.3 of [NF EN 62304] indicates that manufacturer shall include in MMR requirements (implemented to take into account hardware failures and defects of software) according to MD. No details are provided for implementation of this requirement.		
Proposed improvement	Software shall perform self-monitoring to monitor equipment and behavior of software during operation at specified time intervals. Items that can be controlled are: - Random failures of hardware components, - Memory areas containing code and invariable data to detect unanticipated changes, - Software behavior errors (treatment differences, data corruption, data consistency control...), - Data transmission… In case of failure detection, software shall give appropriate responses on time (set to fallback position, degraded operation...).		

Source	[EN 50128] ☐	[ISO 26262] ☐	[IEC 60880] ☒ §6.2, Appendix B.3
	Incidents ☒ (13)	Other ☐	
Justification	During operation this kind of control allows to detect early failure which can come from: - Hardware, - Equipments interfaced to software, - Software. In addition, poor or no development of self-monitoring is primary cause of 13 incidents. This explains why the level of importance is major. Contribution of manufacturer is considered medium because self-monitoring has to be implemented and tested.		

5.3.3.2.Security part

5.3.3.2.1.Introduction to the concept of Security

Before proposing normative improvements to Security of Software, it is necessary to introduce concept of Security.

The term « security » denotes here all processes, means or techniques implemented to ensure following properties of software:

- Confidentiality: ensuring that information is only accessible by person or entity to which access is authorised,
- Availability: Ability of software to operate at request,
- Integrity: That neither data nor software have been modified, altered, or removed during treatment or communication.

Achievement of these properties is to make any malicious attack more difficult, and thus to prevent data compromise and misuse of Medical Device.

Proposed improvements are classified in two categories:

- requirements during development process and during lifecycle of the software
- requirements advocating protection related to software security and development environment

5.3.3.2.2. Security strategy

Type	Creation	Level of importance	Major
		Contribution	3
[NF EN 62304]			
Finding	[NF EN 62304] contains no requirement for software development process to ensure security from digital malicious acts. Only one requirement is present in §5.2.2 e) indicating that manufacturer shall include security requirements with examples by Note. This explains why the type of proposal of this improvement is "Creation".		
Proposed improvement	Guidelines for Security Strategy (security processes) are to be addressed in [NF EN 62304] but a specific reference source treating this subject is needed. Specific documentation to security shall be implemented: - Realisation of a Threat Prevention Plan. This plan shall: ✓ Assess vulnerabilities associated with development environment. ✓ Assess MD and software vulnerabilities throughout entire software lifecycle. ✓ Evaluate critical functions of MD or software. ✓ Assess threats on Privacy / Availability / Integrity properties according to the evaluated vulnerabilities and critical functions. ✓ State countermeasures and security requirements for all assessed threats. ✓ Be linked to Software Development Plan and Risk Management Plan. ✓ Serve realization of Software Security Report verifying security requirements consideration. - Realization of a Software Security Report. This report shall: ✓ Assess activities related to software security. ✓ Assess consideration of safety requirements issued in Threats Prevention Plan. ✓ Rule on and advise on software security. Threat Prevention Plan and Software Security report shall be integrated in Risk Management analyses ([ISO 14971]). Requirements applicable throughout software lifecycle shall be integrated into activities. This applies: - Software Specification and Design: ✓ Security and countermeasures requirements issued by Threat Prevention Plan shall be included in design and indicated as such. ✓ Access strategies limited to software by users' authentication shall be defined. ✓ Traceability matrix shall allow verifying that countermeasures and security requirements are taken into account in the design.		

<table>
<tr><td></td><td colspan="3">- Software Verification and Validation:
✓ Activity dedicated to verification and validation of the software security is to be carried out.
✓ Effectiveness of protection measures and security functions must be confirmed and / or demonstrated by suitable tests.
✓ Activity of security vulnerabilities detection can be carried out (penetration testing).

- Software Operation and Maintenance:
✓ Users must be aware of security principles to be able to identify abnormal behaviors of MD.
✓ Identify actors involved in the updating of software and their roles. An authentication during update is mandatory.
✓ Describe actions to be realised for software updating.

Furthermore, an audit can be performed to check implementation of safety requirements in the work environment, development process, and other stages of software lifecycle.</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] ☐</td><td>[ISO 26262] ☐</td><td>[IEC 60880] ☒
§5.6, §5.7, §12.2</td></tr>
<tr><td>Incidents ☐</td><td colspan="2">Other ☒
[FDA_Cybersecurity]</td></tr>
<tr><td>Justification</td><td colspan="3">Increase in risk related to IT security and legislation related to confidentiality of medical data require establishment of a process of Software Security. Establishment of a strategy for maintaining a certain level of security throughout software lifecycle is essential.

Level of importance is therefore considered major, and contribution to manufacturer is considered high (process and strategy to define and to apply).</td></tr>
</table>

5.3.3.2.3.Security protection

Type	**Creation**	Level of importance	**Major**
		Contribution	3
[NF EN 62304]			
Finding	[NF EN 62304] contains no requirement for protection to ensure security deal with digital malicious acts.		
Proposed improvement	Guidelines for security protections are to be addressed in [NF EN 62304] but a specific reference source treating this subject is necessary (as previous point). Requirements for software security protections are as follows: - Design Activity: ✓ Actors/users roles and privileges shall be clearly defined. ✓ Design of Security shall not be based on secret of algorithms. Always consider that a hacker can have an access to the code. ✓ Minimization of data used and of complexity of software. ✓ To avoid using software / third party libraries (SOUP). Otherwise, their use shall be justified and a study of their security shall be carried out and considered. ✓ To use appropriate mechanisms for user authentication. Use a combination of the following information: • Login, • Password robust (minimum number of characters, special characters, periodic change of password...), • Control of the number of attempted to enter Login / Password, • Physical media (badge, smart card), • Fingerprint (biometric information). ✓ To define mechanisms to ensure data confidentiality (encryption) during storage and communication. ✓ To establish mechanisms to ensure availability of critical functions even if security is compromised. ✓ All communications must be secured. Following set of known defenses are recommended ([EN 50159]) • Sequence number, • Message Dating, • Elapsed time, • Identifiers of source and recipient, • Return message, • Identification Procedure, • Security code, • Cryptographic techniques. ✓ To use communication protocols recognized as secure. ✓ To use methods of encryption / signing as strong as possible according to the context. ✓ Privileges assigned to users shall be reduced to the minimum		

<table>
<tr><td></td><td colspan="3">necessary to carry out functions dedicated to the associated role.

- Development Activity
✓ Specifics of the language used shall be taken into account (eg : Unsafe C function - vs. strcpy strncpy).
✓ To control entry: Integrity control (checksum: md5, sha...), syntax and semantic data check (if possible), source authentication (signature).
✓ Steps shall be taken against presence of function or hidden paths in software.

In addition, software development environment shall be secure, physically or digitally. Workstations shall be secured:
- Software protection use (Firewall, Antivirus...),
- Data encryption: they mustn’t be read in case of physical access,
- Partitioning of digital network to deter any attack from outside,
- Regulated and secured physical access (badge...).</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] ☐</td><td>[ISO 26262] ☐</td><td>[IEC 60880] ☒
§5.6, §5.7, §12.2</td></tr>
<tr><td>Incidents ☐</td><td colspan="2">Other ☒
[FDA_Cybersecurity]
[EN 50159]</td></tr>
<tr><td>Justification</td><td colspan="3">Risk increase in IT security and legislation related to confidentiality of medical data require establishment of software security protections.
Level of importance is therefore considered as major, and contribution for manufacturer is considered strong because software security protections shall be defined and implemented.</td></tr>
</table>

5.3.4. IMPROVEMENT SYNTHESIS [NF EN 62304]

This table summarizes improvements to [NF EN 62304].

Category	Criteria	Proposed improvement	Significant level / Contribution	Incidents number (1)
[NF EN 62304]				
Process	Organization	Organizational structure	Major / 2	-
		Assessment/Certification	Major / 2	-
	Method	Global traceability	Minor / 1	3
	Documentation	Documents to produce	Minor / 1	-
		Performance and environment constraints	Major / 2	4
		Installation	Major / 1	4
		Detailed design	Major / 3	12
	Maintenance	-	-	-
Development Techniques	General	Development techniques	Major / 2	-
	Architecture	Architectural constraints	Major / 2	5
	Programming rules	Implementation constraints and LU checks	Major / 2	20
	Testing techniques	Tests Techniques	Major / 2	-
		Unit tests	Major / 2	4
	Interface, HW/SW integration	Interfaces et software integration	Major / 3	47
	Tools & SOUP	Tools Qualification	Major / 2	-
		SOUP requirements collecting	Minor / 1	-
	Settings	Software configured by application data	Major / 3	5
Safety & Security Strategy	Safety Strategy	-	-	-
	Safety Protection	Safety Protection – Self-monitoring	Major / 2	13
	Security Strategy	Guidelines for Security strategy	Major / 3	-
	Security Protection	Security Protection	Major / 3	-

(1) : « Number of incidents » corresponds to incidents (identified in the first phase of this study) related to recommendations.

5.4. AREAS FOR IMPROVEMENT TO [NF EN 62366]

[NF EN 62366] concerns application of engineering's ability to a medical devices use.
As indicated in the introduction of the standard, engineering process of usability is intended to achieve ability to a reasonable use in order to minimize user errors and minimizing associated risks for use.
In this standard, suitability for use is limited to characteristics of user interface.

Based on the analysis of incidents carried out in phase 1, 3 proposals for improvement are present in this report.

5.4.1. USER INTERFACE

Type	**Complement**	Level of importance	**Minor**
		Contribution	1
[NF EN 62366] §5.7 : User Interface Implementation			
Finding	Requirements on the user interface (HMI) are missing in the normative part of [NF EN 62366]. Indeed, Section 5.7 of this standard for design and implementation of the user interface is very basic: « The MANUFACTURER shall design and implement the USER INTERFACE as described in the USABILITY SPECIFICATION utilizing, as appropriate, USABILITY ENGINEERING methods and techniques.»		
Proposed improvement	Introduce user interface requirements in normative part (§5.7), based on information provided in: - Table D.3 of §D.2.6. This table identifies examples of user interface requirements concerning: ✓ Generalities related to edition in the HMI, ✓ List boxes display (differentiation of selected value, choice without scrolling...), ✓ Menus (upper left corner reserved for the alarm off indicator...), ✓ Display (luminance, contrast…), ✓ Control devices (screen control panels, spacing between buttons...). - §D.4.6.3 « Software User interface». This chapter is included in the informative Appendix D. It identifies requirements that may be included in a user interface. For example, these specifications may include: ✓ screen and window layouts including labelling, fonts, use of colour, and graphics, ✓ dialog flow, including audible events, ✓ description of expected USER interaction with displays and controls. - Table G.7 « Respond to and inactivate ALARM SIGNALS ». This table gives design information regarding alarm signals: ✓ Detectability: remotely audible and visible alarm,		

	✓ Understanding: Clarity of messages, distinction between alarm message and informative message, priority management, understanding of alarms priority… ✓ Acknowledge: Identification of alarm's cause, action required to resolve alarm's cause….		
Source	[EN 50128] ☐	[ISO 26262] ☐	[IEC 60880] ☐
	Incidents ☒ (3)	Other ☐	
Justification	This improvement aims to add requirements in normative part of [NF EN 62366] for manufacturers to define their software user interfaces. A poor ergonomics due to a lack of clarity in the presentation of displayed items is primary cause of 3 incidents. For information, these three incidents focus on LAP category of software (LAP is not a MD software but is taken into account in the context of this study). Degree of importance is considered minor, and contribution to manufacturer is low because this activity is already performed by most of manufacturers.		

5.4.2. USER INSTRUCTIONS

Type	**Complement**	Level of importance	**Major**
		Contribution	1
[NF EN 62366] §6 : Accompanying Document			
Finding	Chapter 6 of [EN 62366] does not indicate any constraint towards provision of software operating instructions (Accompanying Document) with a user interface, even if it identifies constraints when instructions use is provided.		
Proposed improvement	Provision of software operating instructions is required. In addition, requirements on formalization of an operating manual are to be added to §6 of [NF EN 62366]. This applies: - Constraints related to application data (settings), - Perimeter of use, - Limits and restrictions on use.		

Source	[EN 50128] ☐	[ISO 26262] ☐	[IEC 60880] ☐
	Incidents ☒ (11)	Other ☐	
Justification	This improvement aims to add requirements in [NF EN 62366] related to formalisation of user manual so that user has all information for a proper use of software if he is trained to use it (see improvement proposal on training). 11 incidents are due to incompleteness in user manual. For information, these 11 incidents focus on software categories: - 4 incidents LAP (LAP is not a MD software but is taken into account in the context of this study), - 4 incidents RT, - 2 incidents IMA, - 1 incident LABM. Thus, level of importance is considered major. Contribution for manufacturer is low because this activity is already performed by most of manufacturers.		

5.4.3. USERS TRAINING

Type	**Complement**	Level of importance	**Major**
		Contribution	2
[NF EN 62366] §7 : Training/Training Materials			
Finding	Chapter 7 of [NF EN 62366] applies for a training specific MD if it is required for safe and effective use of a main function of service by the intended user. Nevertheless, several lacks has been identified.		
Proposed improvement	Chapter 7 of [EN 62366] shall be completed in order to add: - Constraints on training: ✓ Trainer's competence, ✓ Detailed programs of training, ✓ Skills required for user. - Constraints on the content of training by introducing requirements on parts to be contained training. For example: ✓ Functionality of software and how to use them, ✓ Effects of recurring errors (Experience feedback...)…		

<table>
<tr><td></td><td colspan="3">- Constraints on audit of training material: Once training materials established, it is necessary to conduct an audit of its content to ensure it is in accordance with features offered by the software and there is no ambiguity.

- Constraints on the evidence of training - proof that medical staff has been trained and is able to use MD.</td></tr>
<tr><td rowspan="2">Source</td><td>[EN 50128] ☐</td><td>[ISO 26262] ☐</td><td>[IEC 60880] ☒ §12.4</td></tr>
<tr><td>Incidents ☒ (12)</td><td colspan="2">Other ☐</td></tr>
<tr><td>Justification</td><td colspan="3">This improvement aims to improve user's knowledge about MD software, and thus to have more confidence in his mastery.

12 incidents are due to incompleteness or inconsistency in forming MD software user.
For information, these 12 incidents focus on the following software categories:
- 5 incidents LAP (Non DM software, but considered in the context of this study),
- 2 incidents RT,
- 3 incidents IMA,
- 1 incident LABM,
- 1 incident SILBM.

Thus, level of importance is considered major.
Contribution for manufacturer is medium, because this improvement requires a formalization of training process (content and content verification).</td></tr>
</table>

5.4.4. SUMMARY OF IMPROVEMENTS [NF EN 62366]

This table summarizes improvements to [NF EN 62366]:

Proposed improvement	Significance Level / Contribution	Number of Incidents (1)
[NF EN 62366]		
User interface	Minor / 1	3
Manual of use	Major / 1	11
Users training	Major / 2	12

(1) : « Number of incidents » corresponds to incidents (identified in the first phase of this study) related to recommendations.

5.5. AREAS FOR IMPROVEMENT OF [ISO 14971]

5.5.1. INTRODUCTION

[ISO 14971] concerns application of Risk Management to Medical Devices related to patient, operator or other persons / equipment.

This standard identifies a process to enable manufacturer to identify hazards and hazardous situations associated to medical devices, to estimate and assess risks, to control these risks and to monitor effectiveness of this control.
Requirements of this International Standard apply to all stages of lifecycle of a MD.

This standard has a manual [TR 80002-1] in the form of a technical report. This guide contains further details about software specificities for a better understanding of standard in a software perspective (See [TR 80002-1], Introduction).

In a general way, [ISO 14971] and its guide [TR 80002-1], which is a very detailed software specifications supplement, enable to describe risk management process in a sufficient way. However, a proposal for improvement has been identified.

5.5.2. SOFTWARE SECURITY

Type	Creation	Level of importance	Major
		Contribution	3
[ISO 14971] (creation of a dedicated chapter)			
Finding	There is no requirement about strategies and protections related to software security in [ISO 14971].		
Proposed improvement	Strategies and protections related to software security have to be defined. See following detailed files: - Security strategy - chapter 5.3.3.2.2, - Security protection - chapter 5.3.3.2.3 Guidelines for the strategy and protections of security are to be addressed in [NF EN 62304]. Risk management related to security issue is to be included in [ISO 14971]. Treatment of security issue as a whole is to be achieved in a specific reference source.		
Source	[EN 50128] ☐	[ISO 26262] ☐	[IEC 60880] ☒ §5.6, §5.7, §12.2
	Incidents ☐	Other ☒ [FDA_Cybersecurity] [EN 50159]	
Justification	This is a major point of improvement.		

5.5.3. SUMMARY OF IMPROVEMENTS [ISO 14971]

This table summarizes the improvements to [ISO 14971]:

Proposed improvement	Significance Level / Contribution	Number of Incidents (1)
[ISO 14971]		
Security strategies and protections	Major / 3	-

(1) : « Number of incidents » corresponds to the identified incidents from the materiovigilance statements analyzed in the first part of the study in relation to the recommendations.

6. APPENDIX 1 - PRESENTATION OF STANDARDS USED FOR PHASE 3

6.1. EN 50128

Aim
[EN 50128] provides a set of requirements for development, deployment and maintenance of any software for railway control and protection systems.

Field
Railway

Definition of safety levels
[EN 50128] defines five Software Safety Integrity Levels (SSIL 0 to SSIL 4) according to the risk level linked to the use of the software in the subsystem:
- SSIL 0: not safety related
- SSIL 1: low
- SSIL 2: middle
- SSIL 3: high
- SSIL 4: very high

These safety integrity levels are used to define methods apply to each phase of software lifecycle.

Description
[EN 50128] is applied to any safety software for railway control and protection systems (onboard and ground systems).
This standard can be used for embedded software in railway systems but also in external control systems (not embedded).
This standard applied also to « generic » software (usable for various applications) and their associated settings, and integration of COTS software. It describes:
- organisation and management of software development, staff skill, lifecycle issues and documentation,
- software assurance,
- software development process,
- application data development process,
- software deployment and maintenance,
- methods to implement for each phase of software lifecycle according to its safety integrity level.

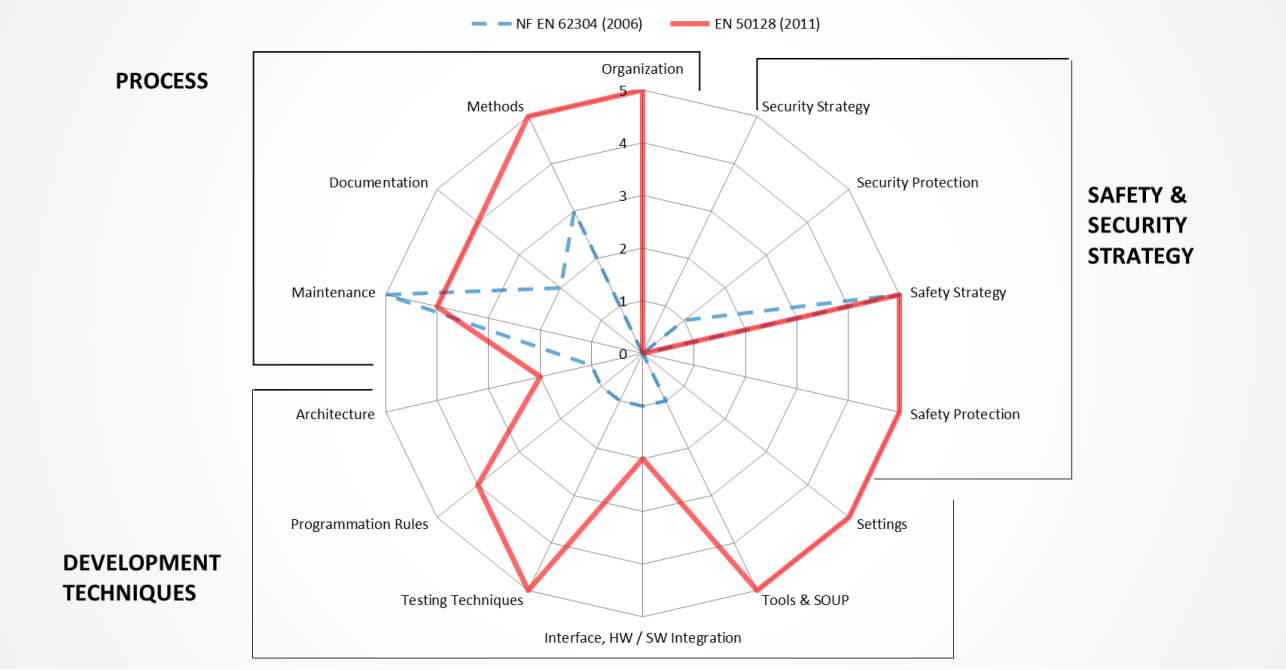
Railway standard does not only covers software with a very high level of safety (SSIL4), but also many of software whose safety level (SSIL1 / SSIL2) can be compared to those in the medical field. In that sense, the standard provides many useful details.
As in other sectors, this standard does not address at all the security aspects. However, information are available particularly in [EN 50159] - Part 2 which deals with the safety communication in open transmission systems.

Some examples of applications:
- Emergency braking of train,
- Door access control software,
- System called « dead man »,
- Driver HMI with speed information,
- Railway signaling software,
- Onboard-ground management software,
- Centralized control station (external control systems),
- Generic software and associated settings.

Advantages towards [NF EN 62304]:
[EN 50128] has the following advantages:
- It was designed on the basis of a strong industrial experience to meet the needs of the sector in terms of functional safety. It has been the subject of a major evolution in 2011, and thus integrates new software development techniques.
- It defines requirements for organizational structure, relationship between organizations and distribution of responsibilities involved in development, deployment and maintenance activities.
- It is very detailed on the process to be applied throughout software lifecycle.
- It describes methods to be applied depending on the safety integrity level for each phase of software lifecycle.
- It defines programming rules and safety protection to implement.
- It addresses test part in its entirety.
- It defines requirements for tools used in the development process and classified into three categories.
- It defines in detail requirements for configurable software.

Disadvantages towards [NF EN 62304]:
[EN 50128] has no lacks towards [NF EN 62304]. In contrast, some aspects as security are not addressed.

[NF EN 62304] vs [EN 50128] radar diagram:



6.2. ISO 26262- PART 6

Aim
[ISO 26262-6] defines requirements for the development of software having to meet safety constraints in the automotive field. To be more complete, this chapter will also be based on other parts of [ISO 26262].

Field
Automotive.

Definition of safety levels
[ISO 26262] defines four safety integrity levels for automotive (ASIL A to ASIL D). This expresses the required safety level of a system function that must be respected in the system design and development, level A being the lowest and level D the highest. If the risk does not affect safety, it takes level QM (Quality Management)…

Description
[ISO 26262] - Part 6 applies to any integrated software (embedded) in a vehicle and participating in a safety function. Vehicles incorporate more and more electronics and software to ensure safety functions.

[ISO 26262-6] establishes requirements for:
- specification of the software requirements and software safety requirements,
- software architecture,
- software implementation,
- software testing,
- software integration,
- software verification.

Methods to be applied depending on the safety integrity level of the software are defined, for each phase of the software lifecycle.

Automotive standard is one of the newest, and focuses on aspects related to functional safety. It covers software whose safety levels are generally similar to those in the medical field. It also defines a suitable process for so-called "configurable" software, and adresses development of techniques including architecture and Hardware / Software interface.

Standard does not address or few security aspects.

To address this topic in the automotive field, work has been launched to meet the increasing risk of malicious attacks. Initiatives such as the research program EVITA (E-Intrusion Protected Vehicle Safety Applications) were launched in 2009 to develop a reference source for design, verification and development of a safe architecture (Safety & Security) for automotive embedded systems.

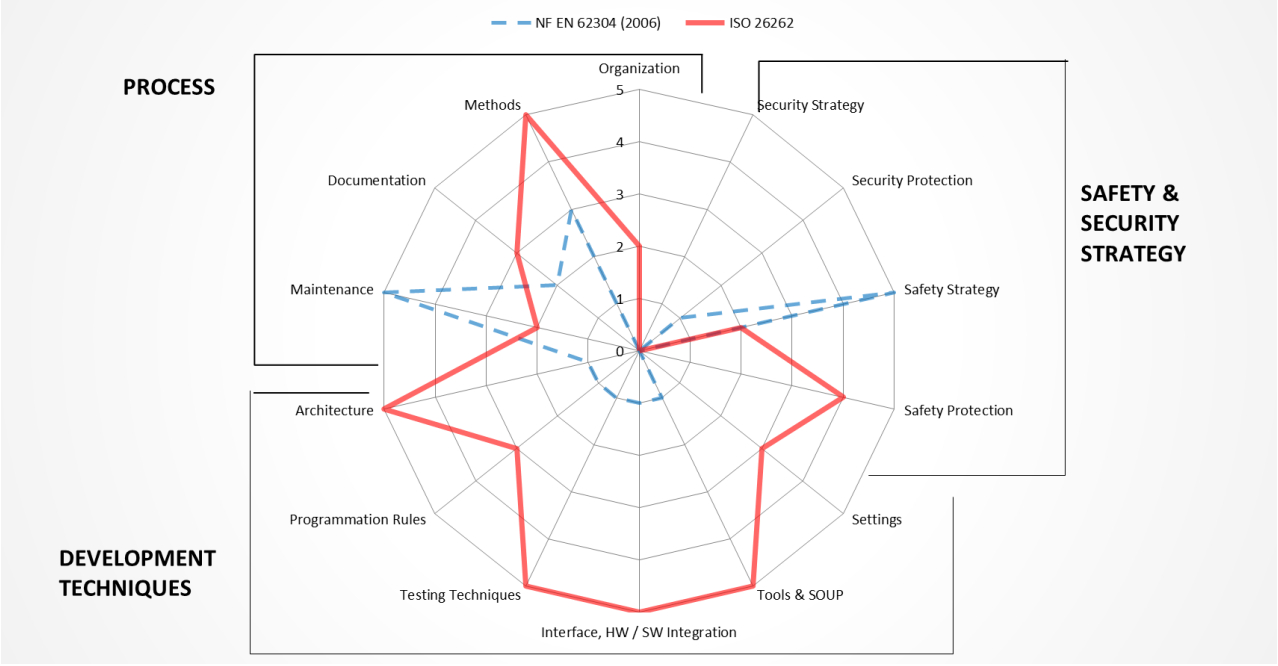
Note: The useful parts for comparison analysis with [NF EN 62304] which would not be a part of [ISO 26262-6] but present in other parts of the [ISO 26262] will be taken into consideration in the comparison activity.

Some examples of applications:
- ADAS Advanced Driver Assistance System (involved in various functions such as automatic parking assistance, blind spot detection...),
- Engine management,
- Airbags management,
- Electric Power Steering,
- Battery Management System,
- Opening / closing of the doors,
- Parking brake,
- Opening / remote control.

Advantages towards [NF EN 62304]:
[ISO 26262-6] has the following advantages:
- It provides a clear and structured framework for the phases of the software lifecycle.
- It provides a complete and concise vision, basing itself on other parts of [ISO 26262].
- It takes into account the software development environment and material constraints.
- It describes methods to apply according to the safety integrity level for each phase of the software lifecycle.
- It defines the programming and protection safety rules to implement.
- It addresses the test part in its entirety.
- It defines requirements for tools used in the development process.
- It provides very detailed guidance on architectural choices, and information to provide regarding Hardware / Software interface.

Disadvantages towards [NF EN 62304]:
[ISO 26262-6] has the following disadvantages:
- It gives few details on software maintenance.
- It gives few details on safety strategy applied to the software.

[NF EN 62304] vs [ISO 26262-6] radar diagram:



6.3. IEC 60880

Aim
[IEC 60880] defines principles and requirements for safety software of nuclear power plants.
This Standard is applicable to high reliability software for computer-based systems performing category A functions and used for instrumentation and control systems important to safety.

Field
Nuclear

Definition of safety levels
[IEC 60880] concerns software of Class I performing Category A and/or B and/or C and/or unclassified safety functions.

Categories are defined in [IEC 61226] and classes are defined in [IEC 61513].

[IEC 61513] standard defines class of important I & C systems for safety:
- I & C Class 1 systems are primarily designed to perform category A functions, but can also perform category B and/or C and unclassified functions;
- I & C Class 2 systems are primarily designed to perform category B functions, but can also perform category C and unclassified functions;
- I & C Class 3 systems are mainly designed to perform category C functions, but can also perform non-classified functions.

Description
[IEC 60880] provides requirements for each stage of the software lifecycle, from specification to operation. Software verification and modification process are also addressed. In addition, standard devotes a chapter to the common cause failures, tools and their qualification.

Advantages towards [NF EN 62304]:
Criticality of the software covered by this standard may be considered far superior to that of MD software. Nevertheless, this standard provides interesting notions on the following points:
- It addresses the issue of security, which is the ability to prevent access or unauthorized modifications (malicious) of data and/or programs.
- It treats the means of defence against common cause failures.
- It addresses test part in its entirety.
- It defines programming and protection rules to implement security.
- It introduces problems related to the tools qualification.

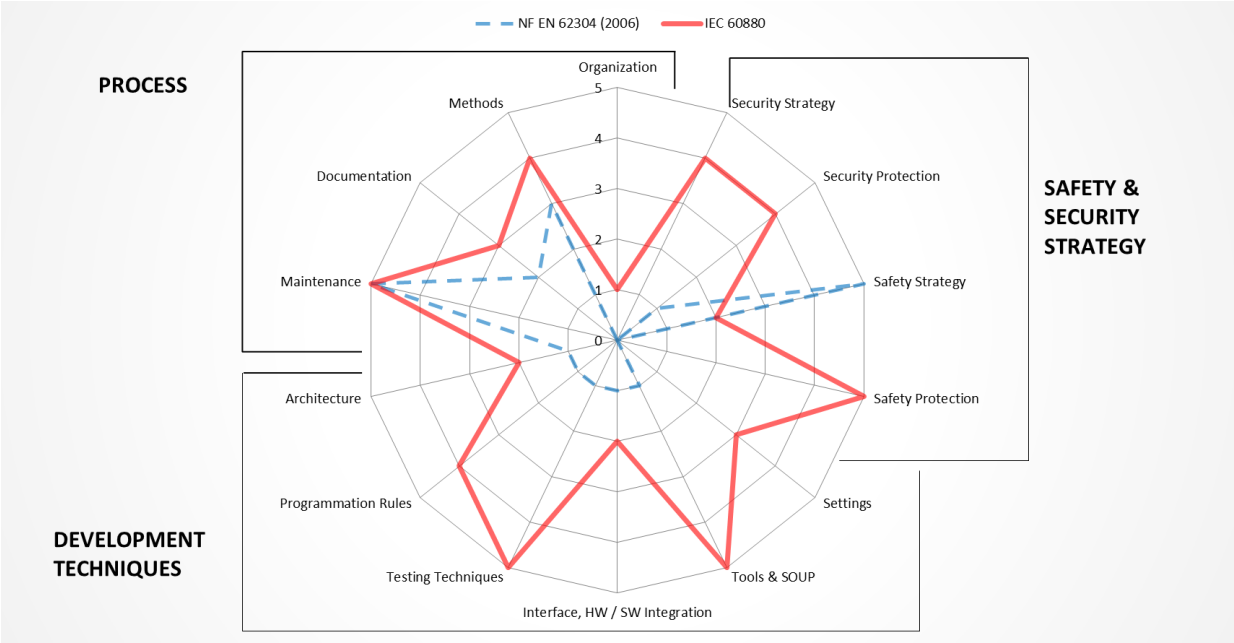
In addition, the standard addresses the issue of operators training.

Disadvantages towards [NF EN 62304]:

[IEC 60880] has the following disadvantages:

- It does not address the need for full traceability.
- Security strategy is not developed enough.
- It uses the wording « it is appropriate that », implicitly considering requirement as recommendation, and not as obligation.

[NF EN 62304] vs [IEC 60880] radar diagram:



7.APPENDIX 2 - DEFINITION OF CRITERIA AND NOTES

7.1. CRITERIA

Criteria are classified into three categories. Criteria used for comparison are as follows:

PROCESS:

- **Organization**: Information related to the organization of software development to implement and entities or persons involved in the software lifecycle.
    - Role: Information on responsibilities of stakeholders (Example: Project Manager, Designer, Audit Manager, Validation Officer...).
    - Responsability: Information on responsibilities of stakeholders in the development:
        - Example of one of the responsibilities of a designer: Must transform specified requirements into acceptable software solutions.
    - Skill: Information on necessary skills for each role
        - Example of one of the Project manager skills: he must master standards defining the software development process.
    - Independence: Information related to the independence between entities
        - Example: The same person must not combine « Responsible for validation » and « designer » roles.
- **Methods**: Recommendation of methods apply at each stage of the software lifecycle based on the software safety level.
    - Lifecycle: Information on a lifecycle to be adopted (V cycle, cascade Cycle...).
    - Description of activities of the descending part of the lifecycle.
    - Presentation of verification activities: Review, proofreading sheet, traceabilities verification, verification team…
    - Presentation of validation activities: Unit testing, integration testing, and validation testing.
    - Traceability: Information on traceability to implement during the software development.
- **Documentation**: Recommendations on the documentation format and content produced during the software lifecycle:
    - Document: List of documents required for each phase
    - Contents: Information on the content of each document (Chapter predefined, content formalization method...).
- **Maintenance**: Information on software maintenance.
    - Impact analysis: Information on the impact analysis of an application for software changing or modification.
    - Non Regression: Information on the non regression of the activities after taking into account the changing/modification.

TECHNIQUE FOR DEVELOPMENT

- **Architecture**: Information on safety architecture choices and representation:
    - Representation via the Modeling: Formal/semi-formal method, structure diagrams (class, object...), behavioral diagrams (activities, state machines)…
    - Safety architecture definition. Example:
        - Detection/Error Handling,
        - Redundancy: Parallel processing of channels,
        - Diversification: Software function or Software performed N times in different ways.

- **Programming rules** (design/coding): Constraints on design and programming rules:
  - Obligation to make a programming manual: Information on the programming manual combining the coding, naming and good practices rules.
  - Introducing adequat programming language: List of programming language to use or not, according to the software safety class.
  - Static code analysis.
- **Testing techniques**: Information about testing techniques
  - Constraints on the test coverage: Information on the coverage that tests need to achieve (functional, structural, durability, performance testing...).
  - Constraints on testing techniques: Information on the testing techniques (equivalence class, boundary testing...) and their environment.
  - Constitution of test datasets: Test scenarios to replay each delivery, grouped in a tests catalog or specification.
- **Interface, Hardware / Software integration:** Information on the definition of the interface and integration between Hardware and Software
  - Definition of HW / SW interfaces in terms of container and content, compliance with material constraints (clock, datasheet...).
  - Information on assembly process of equipment and software items to check the compatibility of the software in its physical environment.
- **Tools & SOUP**: Information about tools used in software development:
  - Choice of tools.
  - Qualification and validation: Information on qualification tools and their validation.
  - Constraints on the selection and management of COTS and SOUP.
- **Settings:** Information on configurable software. These software can be adapted to the requirements of a specific application through parameters, also called application data or configuration data.

SAFETY & SECURITY STRATEGY

- **Safety protection**: Information on software protection techniques and fault tolerance. Example:
  - Fault Detection & Diagnosis,
  - Critical data protection,
  - Detection codes and error correction,
  - Set to fallback position / degraded mode.
- **Security protection**: Information on software protection techniques against unauthorized and intentional actions (malicious actions). Example:
  - User access permissions,
  - Authentification (password, key, fingerprints…),
  - Confidentiality,
  - Integrity of exchanges.
- **Safety strategy**: Information about the software analyzes demonstrating the safety level against unpredictable events.
  - Achievement of a document demonstrating safety,
  - Risk Analysis: Information on potential risks analysis for the software safety, and the safety requirements identification,
  - Dysfunctional analysis as AEEL,
  - Common mode failures analysis.

- **Security strategy**: Information on analysis demonstrating software security level against malicious, unauthorized and intentional actions.
    - Achievement of a document demonstrating security,
    - Security Analysis: Information related to the analysis of potential threats to software security.

7.2. NOTE

At each of the criteria defined in the previous chapter, a note (which can be found on the radar axes) from 0 to 5 was attributed:

- **0**: Absence of the criterion in the standard.
- **1**: Criterion mentioned.
- **2**: Brief explanation of the criterion.
- **3**: First level of detail of the criterion.
- **4**: Interesting but incomplete level of detail, or all the criterion elements are not present.
- **5**: Full explanation with a high level of detail of all the criterion elements.

If criterion element is missing, note can not be 5, even if explanations and level of detail are very high for the other elements.