Product Service

Choose certainty.
Add value.

Med-Info

International expert information
for the Medical Device industry

Medical Device software

Background

Software differs from other parts of Medical Devices. It is intangible, has a significant number of internal states which make it highly complex, and failures in software are systematic failures (by definition). Dealing with these challenges requires different approaches (as compared to hardware) regarding the processes to be used during the development and the maintenance of software when software is used as stand-alone software or when it is part of a Medical Device.

Now there is a standard for software life cycle processes (EN 62304:2006). The goal of the standard is to create a common framework for the software life cycle with regard to processes, activities and tasks. These processes and tasks are necessary for the safe design and maintenance of Medical Device software. The documentation and the test efforts are scaled according to the criticality of the software.

What standards exist?

- EN 60601-1-4 (Medical electrical equipment – programmable electrical medical systems)
- EN 60601-1, 3rd edition (Medical electrical equipment – general requirements for basic safety and essential performance), clause 14
- EN 62304 (Medical Device software – software life cycle processes)

Harmonization status

Europe: EN 62304:2006 has been harmonized since November 2008 under MDD (93/42/EEC), AIMD (90/385/EEC) and IVDD (98/79/EC). A transition period does not exist.

USA: IEC 62304:2006 has been an FDA-recognized consensus standard since March 2006.

EN 62304:2006 therefore applies to all kinds of software within the range of Medical Devices regardless of their criticality.

What does the standard require?

The new standard defines requirements for five processes. These requirements have to be implemented within the quality management system of the company.

1) Software development process

This process defines requirements to be followed during software development. It comprises the subprocesses software development planning, software requirements analysis, software architectural design, software detailed design and implementation. For each of these phases a corresponding test level has to be defined. The development effort (and related documentation) depends on the criticality of the software.

2) Software maintenance process
This process defines what has to be considered during the maintenance of the software.

3) Software risk management process
Risks related to software are sometimes different from those related to hardware. This chapter defines requirements on how to perform a risk analysis on the software. For risk management and software other guidance papers are available or currently under development:
AAMI TIR32:2004 and IEC/TR 80002-1 Ed. 1.0.

4) Software configuration management process
The software itself, the associated documentation, and the tools used (e.g. compilers, libraries) have to be subject to configuration management.

5) Software problem resolution process
Whenever software is changed after a certain phase, the changes have to be documented and accepted, and the impact of the changes has to be analyzed (e.g. regarding documentation and tests already performed).

Apart from these five processes, the standard contains requirements regarding software classification and commercial software or software for which no adequate documentation exists.

Software safety classification: A safety class has to be assigned to each software system: C = serious injury or death possible; B = no serious injury possible; A = no injury possible. It is possible to break down a software system into components (items), each with its own safety class. In this context, a rationale for the individual safety classification is needed.

Commercial software: The use and selection of commercially available software has to be considered carefully. Potential hazards resulting from commercial software (like operating systems, databases and drivers) have to be evaluated. Since the quality of the development life cycle is typically unknown, software errors have to be assumed. Certain combinations of software with a higher safety classification and commercial software could be critical and need special consideration concerning the associated risks: a software system classified as "C" might not run safely on an off-the-shelf operating system for which no adequate verification and validation can be demonstrated.

Segregation: It is allowed to assign separate safety classifications to individual software items. In this connection, a rationale for the individual safety classification is needed. The rationale for the separate safety classification should include a consideration of the potential side effects of the software items on the safe operation of other software items, and demonstrate that there is sufficient segregation (a software item of class A should not negatively influence software items of a higher safety class).

What does all this mean for you as a manufacturer?

The requirements of the standard have to be implemented within your quality management system. During product design, the challenges regarding software safety classification, commercial software and segregation have to be addressed. The requirements have to be considered for all new or modified Medical Device software.

How can TÜV SÜD Product Service help you?

TÜV SÜD Product Service has a comprehensive service offer:

- Training regarding the requirements and challenges of the standard
- Gap analysis of existing processes and documentation regarding IEC 62304
- Functional safety testing: address the challenges of software safety classification, commercial software and segregation with a functional safety assessment

Your contact partner at TÜV SÜD Product Service can provide further information.

Olaf Teichert
Phone: +49 89 5008-4156
Email: olaf.teichert@tuev-sued.de

Dr. Andreas Purde
Phone: +49 89 5008-4203
Email: andreas.purde@tuev-sued.de

Med-Info March 2013/11 EU