Software Verification and Validation
The Role of IEC 60601-1

Anura Fernando

About the Author

Anura Fernando is a research engineer at Underwriters Laboratories (UL) in Northbrook IL. E-mail: Anura.S.Fernando@ul.com

When biomedical engineers begin to conceptualize a new medical device, verification and validation (V&V) is usually at the forefront of their thoughts. They want to start the effort well by making sure that they have the right tools and processes in place for "building the product right," and they want to be able to conclude their efforts by demonstrating that they've "built the right product" for their customers. Standards have always played a role in V&V, but part of the third edition of IEC 60601-1—the standard on medical electrical equipment—deals with the V&V of increasingly complex medical devices in a whole new way.

The complexity of medical devices has grown in almost unfathomable leaps and bounds from the likes of primitive tools used for trepanation, as shown in Figure 1a, a painting by Hieronymus Bosch circa 1488-1516, to the likes of modern computer-to-brain interfaces, such as BrainGate (Figure 1b), developed in 2008 by Cyberkinetics and Brown University.

Figure 1a. Trepanation, the process of cutting a hole in the skull

Figure 1b. BrainGate

Arguably one of the most influential technologies in driving system complexity has been software. Software can introduce a level of product capability that begins to approach that of the human brain, and product complexity that begins to parallel that of human thought processes. If not designed with great care, software can also induce system failures, such as erratic operation or incorrect processing of data, which have uncanny parallels to conditions of the human brain, such as schizophrenia and bipolar disorder.

While many debate the definition of software, the medical device community has agreed that a software product, which may by itself be considered a medical device, is a "set of computer programs, procedures, and possibly associated documentation and data."[1] Just as human physical and emotional health are intertwined and must be managed together, as system complexity grows, managing medical device complexity holistically becomes paramount to ensuring system health. Addressing both these systematic flaws and random faults (e.g., short-circuit caused by conductive pollution) becomes one of the primary means of ensuring that the medical device will do what it is intended to do and not do that which is not intended.

As stated in its introduction, the third edition of IEC 60601-1 has changed in some very fundamental ways. First, the concept of "safety" has been broadened from the "basic safety"

considerations in the first and second editions of IEC 60601-1 to now include "essential performance," (e.g., the accuracy of physiological monitoring equipment). Second, in specifying minimum safety requirements, a provision is made for assessing the adequacy of the design "process" when this is the only practical method of assessing the safety of certain technologies, such as programmable electronic systems.[2] Finally, the organizational structure of the standard itself has changed in a manner that reflects the importance of software V&V in the context of the overall product. Rather than existing as a "collateral" standard (formerly IEC 60601-1-4), software requirements are now embedded directly into the general requirements of the third edition of IEC 60601-1 via clause 14. This structural change establishes a clearer path for addressing the potential role of software in mitigating basic safety issues, such as fire and electric shock, as well as addressing the role software plays in essential performance (i.e., the safety-related behaviors or functional capabilities of the medical device).

Through these changes in the standard, we begin to see that managing complexity really has to do with managing product development processes in order to minimize systematic defects that could have gone undetected with traditional product validation approaches. It is probable that the functional complexity could result in a tremendous number of test cases that would need to be executed in order to cover all operating states of the device under combinations and permutations of both normal and abnormal conditions. Focus on product development processes can lead to improved testing efficiency by driving toward the identification of key subsets of all the possible test cases using standards driven risk management processes such as those of ISO 14971, *Medical Devices—Application of risk management to medical devices,* per its reference in the third edition of IEC 60601-1.

As identified in Figure 2, among the first steps in managing risk are to identify the hazards and evaluate the risks. It is during these initial steps that a well-informed systems engineer can begin utilizing the guidance provided in these standards to assist with the product development process. Knowledge of the hazards and risks can be included along with such items as customer needs, intended uses, and constraints imposed by available technology, as inputs for requirements specification. Not all hazards can be eliminated by design and thus some may remain inherent to some extent. This residual risk must be assessed and accepted or mitigated by the product developer.

In order to determine the acceptability of residual risk levels and develop appropriate mitigation strategies, several techniques may be used for risk analysis, each with its own pros and cons. The following are examples:[4]

- Using a checklist is one approach to risk analysis, but relies on trial-and-error history and corporate memory to generate a sufficiently comprehensive list. Checklists can guide thinking in all life-cycle phases, but can be problematic if used without careful consideration of each item on the list.

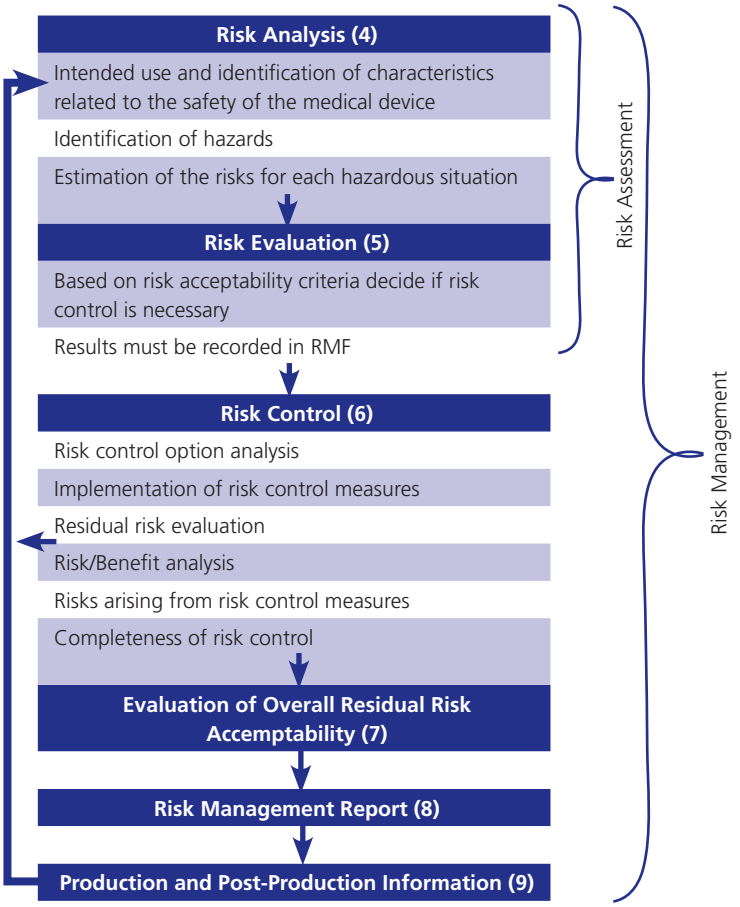

Figure 2. Risk Management Process[3]

- **Fault Tree Analysis (FTA)** is a means of identifying potential causes of hazards. It consists of system definition, fault tree construction, qualitative analysis, and quantitative analysis. It requires foreknowledge of the system, and caution must be exercised to prevent oversight of critical paths due to oversimplification of system representation.
- **Failure Modes, Effects, and Criticality Analysis** is useful for discrete failures. It can be used to establish the overall probability that the product will operate without a failure for a specific length of time or for a specific length of time between failures. This technique, although comprehensive, can be burdensome because of the need to exercise each failure mode of the device under evaluation.

An early philosophy in the analysis of software-related failures was that "software does not fail"; rather, programmers have either been misled with incorrect specifications to develop the software, or the software has been incorrectly implemented.

The current thinking, as reflected in today's standards, deals not just with the systematic aspects of software, but also with the "embedded system" as a whole (that is both software and hardware), as is now seen in IEC 60601-1 clause 14 requirements for the programmable electrical medical system (PEMS) that strive to encompass both "implementation" (systematic) and "nonimplementation" (random hardware) defects in the system. Awareness

of and use of such standards early in the software development lifecycle may lead to improved development by techniques including enhanced static testing (i.e., code walkthroughs, code inspections), dynamic testing (i.e., boundary condition, range testing), and formal testing (i.e., mathematical proof, control modeling).[5]

Static testing may be used to uncover such defects that have occurred, for example, in NASA's Mars Climate Orbiter, where miscalculations of thrust for trajectory adjustment were attributed to inconsistency in measurement units, with some (ground) software using metric units of impulse (Newton-seconds) while in-flight output was specified in English units (pound-seconds). Other defects potentially detected by static testing include: algorithm/logic/processing defects (e.g., "off-by-one," return codes, overflow/underflow), data defects (e.g., pointer errors, indexing, initialization), and system errors (e.g., stack control, version control, resource sharing).[7]

Thus, the third edition of IEC 60601-1 may serve as a tool that facilitates third-party certification to ease market access, but it also has significant utility as a product development tool that brings risk management into the very first stages of the product development process.

Dynamic testing, within the context of a structured development lifecycle, can be used to expose defects similar, in nature, to those seen in the case when, in 1988, the USS Vincennes' Aegis radar system software operated as intended with respect to aircraft identification of an Iranian airliner, but was functionally deficient with respect to the GUI or human factors design—leading to the loss of the lives of all 290 passengers onboard the airliner.[8] Other defects that may be exposed by this type of testing include incorrect control flow, re-entrance errors, data synchronization errors, task synchronization errors, and instrumentation problems.

The types of defects that may be exposed by formal testing parallel and, in some instances, overlap those of the previous two testing methodologies. Formal testing can be used to demonstrate via formal methods (i.e., topology and set theory) that all requirements have been implemented, it can be used to demonstrate system stability, and it can be used to demonstrate correctness and completeness of algorithms.

Thus, the third edition of IEC 60601-1 may serve as a tool that facilitates third-party certification to ease market access, but it also has significant utility as a product development tool that brings risk management into the very first stages of the product development process. Regardless of the specific methods used to test a medical device, one of the most important elements of the product manufacturer's risk management file (i.e., the document that presents a "safety case" for the product through its compliance with IEC 60601) is providing objective evidence that all of the V&V plans have been effectively implemented to manage medical device risks in an appropriate manner. As medical device complexity continues to grow into the realm of systems of systems, and as "intended use" becomes defined by emergent system properties rather than discrete device properties, these standards can continue to provide a solid foundation to be used by medical device product developers and manufacturers for medical device risk management. ■

References

1. **IEC 60601-1** *Medical Electrical Equipment—Part 1: General requirements for basic safety and essential performance,* Third Edition. Geneva, Switzerland. International Technical Commission. 2005
2. **IEC 60601-1** *Medical Electrical Equipment—Part 1: General requirements for basic safety and essential performance,* Third Edition. Introduction. Geneva, Switzerland. International Technical Commission. 2005.
3. **BS EN ISO 14971** *Medical Devices – Application of risk management to medical devices.* Second Edition. United Kingdom. BSi British Standards. 2007.
4. **Leveson N.** *Safeware: System Safety and Computers.* Addison-Wesley Publishing, Inc., 1995.
5. **The Institute of Electrical and Electronics Engineers (IEEE).** Std 1059-1993 IEEE *Guide for Software Verification and Validation Plans,* IEEE Inc., 345 E 47th St., New York, New York, 1993.
6. **Fusco J.** Measure Twice, Cut Once, Embedded Systems Programming, *CMP.* October 2000.
7. **Beatty S.** Sensible Software Testing, Embedded Systems Programming, *CMP.* August 2000.
8. **Ganssle JG,** As Good As It Gets, Embedded Systems Programming, *CMP.* January 2002.