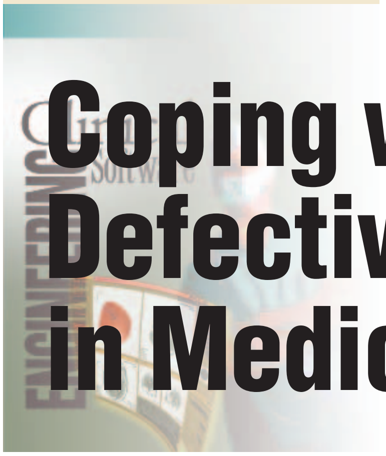
COVER FEATURE



Coping with Defective Software in Medical Devices

Steven R. Rakitin
Software Quality Consulting Inc.

Embedding defective software in medical devices increases safety risks. Given that all software is inherently defective, how can medical device manufacturers identify and manage risk? An effective, tailored risk management process can make the task less daunting.

Software-based medical devices—from defibrillators to electric wheelchairs—have become a critical part of the healthcare landscape. Many medical devices must interface with other equipment, connect to hospital and laboratory information systems, and work in high-stress situations. The increased demands on such devices and their growing pervasiveness have created formidable development challenges for their manufacturers.

Chief among these is ensuring safety, which has become more pressing with the increased complexity of embedded software. Because software engineering is an inherently human process, it is not possible to produce software with zero defects. The challenge for device manufacturers then is to identify and mitigate the risks associated with embedding defective software in their devices.

Medical devices incorporate many types of components. Like a defective electrical component, a defective software component can have dire consequences. However, unlike other types of components, identifying and quantifying the potential effects of defective software components is more difficult. First, as complexity increases, so does the number of defects. Second, because many devices share common components, such as pumps and valves, these components have an established track record. Component manufacturers often provide device manufacturers with performance data for these common components.

Software, in contrast, is often proprietary and developed by (or on behalf of) medical device manufacturers for use only in a specific device. With few exceptions, there is no established track record for software components.

Consequently, it falls to device manufacturers to ensure, to the best of their ability, that software-based medical devices are safe and effective. Meeting this responsibility requires expertise in effective risk management practices, familiarity with software safety, and the adoption of a risk management mind-set.

ELEMENTS OF RISK MANAGEMENT

The risks device manufacturers must address are to patients; operators; third parties, such as service technicians; and the environment. In the US, the Food and Drug Administration regulates the design and development of medical devices, requiring products to be both safe and effective. The FDA's *Quality System Regulation*[1] requires manufacturers to incorporate risk management into their design, manufacturing, and support processes. The "Risk Management: Some Foundational Standards" sidebar lists several international standards that are available to help device manufacturers understand basic risk management principles.

Tailoring the process

The standards encourage device manufacturers to establish processes that are commensurate with the risk their devices present. The notion of "commensurate with risk" is important because it lets manufacturers tailor their development practices according to the estimated or known risks of a particular device or device type. The risk management process to develop, manufacture, and support a defibrillator, for example, would

Published by the IEEE Computer Society

0018-9162/06/$20.00 © 2006 IEEE

be more rigorous than for an electric wheelchair.

Tailoring consists of preparing a device-specific risk management plan that is based on the principles described in the standards. Figure 1 shows one example of this kind of tailoring.

Risk spectrum

For software-based medical devices, the FDA defines a risk spectrum, or *level of concern*, as[2]

> … an estimate of the severity of injury that a device could permit or inflict, either directly or indirectly, on a patient or operator as a result of device failures, design flaws, or simply by virtue of employing the device for its intended use.

According to the same FDA regulation, the level of concern is minor if "… failures or latent design flaws are unlikely to injure the patient or operator." It is moderate if "… a failure or latent design flaw could directly or indirectly result in minor injury to the patient or operator or occur through incorrect or delayed information or through a care provider's action." It is major if "… a failure or latent flaw could directly or indirectly result in death or serious injury to the patient or operator through incorrect or delayed information or through a care provider's action that is based on such information."

Although risk management is required for devices at all levels of concern, documentation and testing are more extensive for devices with a higher level.

Management responsibility

To determine the level of concern for a software-based device, manufacturers must have a deep understanding of their devices and the risks they present. Attaining this insight demands considerable skill and expertise—especially when software is involved.

For manufacturers of software-based medical devices, this means making risk management an organizational core competency. Management must take steps to enhance the skill levels of design engineers, clinicians, service personnel, manufacturing engineers, and quality and regulatory staff, as well as improve the organization's software development and verification and validation processes. All this training and process improvement empowers the risk management team to identify potential hazards early on and implement effective mitigations.

Risk Management: Some Foundational Standards

ISO 14971, Risk Management—Application of Risk Management to Medical Devices, Int'l Standards Org., 2000. Defines several risk management terms and provides a framework for an effective risk management process.

IEC 60601-1-4, Medical Electrical Equipment, Part 1: General Requirements for Safety and Essential Performance, Collateral Standard: Programmable Electrical Medical Systems; ed. 1.1, Int'l Electrotechnical Commission, 2000. IEC 60601-1-4. Defines many basic principles of risk management, including the definition of risk as the combination of probability and severity.

ANSI/ISO/AAMI 13485, Medical Devices—Quality Management Systems—Requirements for Regulatory Purposes, Int'l Standards Org., 2003. Provides the framework for a quality system for medical device manufacturers and requires establishing a risk management process based on ISO 14971 and using it throughout the product realization process.

ANSI/AAMI SW68, Medical Device Software, Software Life-Cycle Processes, Assoc. for the Advancement of Medical Instrumentation, 2001. Defines requirements for a software development life cycle and requires that manufacturers implement risk management throughout the life cycle on the basis of ISO 14971.

AAMI TIR32, Medical Device Software Risk Management, Assoc. for the Advancement of Medical Instrumentation, 2004. Provides guidance on ways to interpret and apply the ISO 14971 requirements for software-based medical devices.

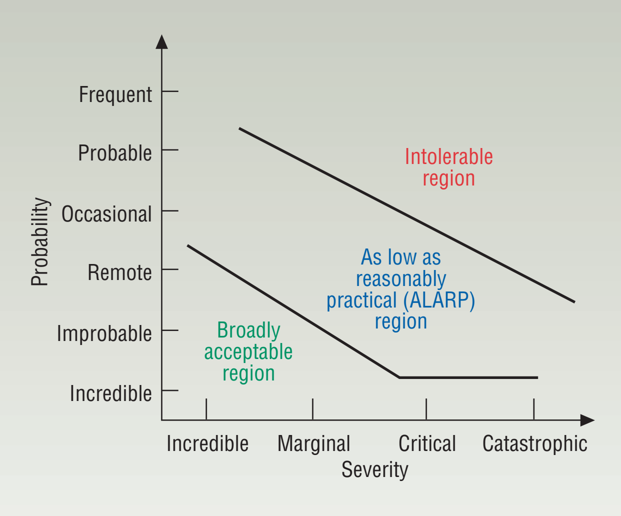


Figure 1. Tailoring the risk management plan. Definitions for probability and severity should be part of a device-specific risk management plan. (From IEC 60601-1-4.)

Software engineers should become intimately familiar with software development practices that are common in the medical device industry by reviewing the standards and FDA guidance.[3] These practices provide both a framework for assessing a device manufacturer's current software development competency and guidance for producing devices with acceptable risk.

Identifying safety-critical software components and data should be part of the risk management plan.

Device manufacturers seeking to improve their software development process can refer to the AAMI SW68 standard (see sidebar) for development life-cycle models applicable to medical device software. The FDA's "General Principles of Software Validation"[3] is an excellent reference for software verification and validation practices typically used in the medical device industry. In addition to the standards listed in this article, the literature has several articles with specific recommendations for software risk management.[4-7]

DO NO HARM

An underlying principle of the medical profession is "First, do no harm." Medical device manufacturers need to embrace the same principle when designing medical devices. Clearly, doing no harm requires the aforementioned management commitment, but it also requires personnel who understand basic software safety[8] techniques.

An example—one of many such techniques—is the use of safe states, which software engineers define on the basis of known failure modes. When the embedded software detects that a failure mode has occurred, it displays information to the operator and puts the device into a predefined safe state to prevent potential harm.

Software engineers need to understand the difference between software reliability and safety. AAMI TIR32 states:

> Reliability is the ability of a system to perform its required functions under stated conditions for a specified period of time. Safety is the probability that conditions (hazards) that can lead to a mishap do not occur, whether or not the intended function is performed. Reliability is interested in all possible software errors, while safety is concerned only with those errors that cause actual system hazards.

Often software is intended to mitigate hazards caused by the failure of other components. For example, in a diagnostic measurement instrument, temperature control is often critical. If the embedded software detects that the temperature is not within the required range, it can display an operator message instead of a measurement result. ISO 14971 requires documented evidence that software mitigations are effective.

In many devices, some software components are clearly more critical than others. Software that calculates a diagnostic result is far more critical than software that prints routine reports with no patient results. Identifying safety-critical software components and data should be part of the risk management plan.

Some software components and data that are likely to be safety critical include[5]

- software whose failure can directly compromise safety requirements (device control software, algorithmic software, measurement software, and so on),
- software used to mitigate failures in other subsystems (memory leak detection software, built-in test software, and so on),
- software that directly accesses safety-critical data,
- support software that the safety-critical software calls,
- any data that the program can display as results,
- data in algorithms or calculations that can lead to displayable results,
- data to determine if a potential hazard might occur, and
- patient demographic data.

Once identified, safety-critical software and data might require additional scrutiny, such as a formal inspection or more extensive testing.

DEVELOPING A RISK MANAGEMENT MIND-SET

Risk management is a cradle-to-grave activity that requires the active involvement of a multidisciplinary team of design engineers, clinicians, service personnel, and quality and regulatory staff. Device manufacturers must make risk management an integral part of their product development process. The postproduction surveillance requirement in ISO 14971 means that even after the device is released for sale, risk management must continue to be part of the corporate mind-set.

During initial development, and especially during maintenance, software engineers need to keep software safety at the forefront of their thinking. AAMI TIR32 captures this idea of everyday risk management:

> Effective software risk analysis and risk management cannot be accomplished in any single meeting or activity. Risk cannot be effectively minimized at the end of the product development cycle by retroactively preparing a software hazard analysis. For software risk management to be implemented properly, a focus on hazard identification, risk evaluation, and risk control must be integrated into each phase and relevant activity of the software development life cycle.

Identifying some common problems and using proven techniques can help device manufacturers meet the daunting challenge of developing a risk management mind-set.

Delaying risk management until device designers have completed the design limits risk mitigation options.

Focus on severity, not probability of occurrence

A common problem when identifying the risks of software-based medical devices is to ignore the inherent differences between risks derived from software components and risks derived from other components.

AAMI TIR32 notes that software risk management might require more focus and different handling. Unlike hardware, software failures are systematic, not random, which means it is difficult to estimate their probability with any accuracy. Added to that is the lack of a software track record.

Thus, the way in which software engineers think about probability of occurrence for a software component is quite different; to be effective, the software aspects of risk management must focus on severity or the risk of harm and not on the probability of occurrence.

Start early

Another common problem is to delay risk management until device designers have completed the design—an approach that limits risk mitigation options. ISO 14971 states that, when manufacturers try to mitigate risks, they should follow three design principles in order of priority:

- Change the design to eliminate risks.
- If following the first principle is not possible, incorporate protective measures in the device or manufacturing process, including the ability to detect conditions that could lead to the risk's occurrence.
- If following the first two principles is not possible, add information in an operator's manual to explain steps to take when conditions that could lead to a risk do occur.

Clearly, these principles emphasize an early start to risk management and thereby give the device manufacturer more flexibility in reducing risk in parallel with device development.

Create a hazards list

ISO 14971 defines a hazard as a potential source of *harm*—physical injury or damage to the health of people (patients, clinicians, and third parties), property, or the environment. The standard requires device manufacturers to identify all known and foreseeable hazards and quantify each hazard's *severity*—the measure of its possible consequences.

Common hazards for specific devices are a useful starting point. Annexes to ISO 14971 can help identify hazards that manufacturers might not have thought to consider. The guiding principle is that if a hazard can physically occur, assume that it will. Again, the focus is not on the hazard's probability of occurrence but on the harm it will cause.

Know the clinical environment

Manufacturers must identify potential hazards related to the clinical environment where the device will be used. For example, devices intended for use in an operating room should be tested in an environment that simulates operating room conditions. Testing in an environment that is as close as possible to the actual clinical environment is an FDA requirement.[1]

Hazards from potential device misuse are also an important consideration, such as those from using the device outside the stated intended use and interference from user-installed software. Interference can occur when the medical device consists of a PC that is running medical device software, and the users install other software on that PC.

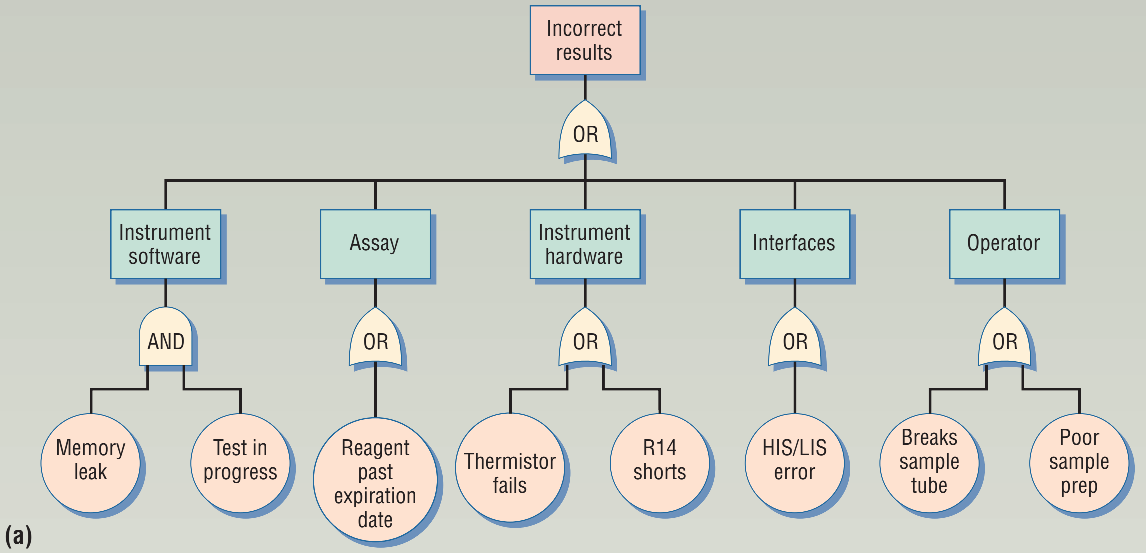
Differentiate failure modes and hazards

Organizations often confuse hazards and failure modes. Failure modes are characterizations of ways that a device can fail. In a typical diagnostic device, for example, a hazard could be "false negative result" while a failure mode might be "fan failure" or "sample contamination." Failure modes can result in hazards, but they do not necessarily represent hazards by themselves. Device manufacturers need to focus on identifying hazards first and then identifying failure modes that can lead to those hazards. *Fault tree analysis* and *failure modes effects criticality analysis* are excellent focusing tools. Figure 2 shows the results of FTA and a FMECA table.

Create a multidisciplinary team

Risk management requires an experienced team that represents a range of appropriate disciplines and skills. A natural tendency is to staff the team with development engineers. But the risk management team also needs the perspectives of manufacturing engineers, service personnel, quality and regulatory staff, and clinicians. Service personnel and clinicians with extensive knowledge of the device's intended use and potential misuse in a clinical setting are key team members.

The team should form at the earliest stages of a development project and remain actively engaged throughout development and manufacturing. It should also have sufficient training in risk management principles



Basic event information			Preliminary risk assessment			Mitigation information			Residual risk postmitigation		
Basic event	Failure mode (causes)	Visible behavior	Severity	Probability	Criticality (risk index)	Mitigation	Verification	Validation	Severity	Probability	Criticality (risk index)
Memory leak	Coding error	Erratic behavior	Critical	Probable	Very high	Code reviews	Code review minutes	No memory leaks detected	Critical	Infrequent	Moderate
Thermistor failure	Age or wear out	Over-heating	Critical	Frequent	Very high	Use MIL-SPEC parts	Verify parts on parts list	Temp. cycling test	Critical	Infrequent	Moderate
Reagent past expiry date	Training	None	Critical	Occasional	High	Training	Training records	Clinical trial results	Critical	Incredible	Low
HIS/LIS error	Coding error	None	Critical	Occasional	High	Code reviews	Code review minutes	No HIS/LIS error detected	Critical	Incredible	Low
Sample tube breaks	Sample prep or defective tube	None	Critical	Occasional	High	Sample prep training	Training records	Clinical trial results	Critical	Incredible	Low
Poor sample prep	Training	None	Critical	Occasional	High	Operator training	Training records	Clinical trial results	Critical	Incredible	Low
R14 shorts	Age or wear out	Display erratic	Critical	Incredible	Low	Mitigation not required …					

(b)

Figure 2. Example analysis for the hazard: Incorrect results. (a) The results of fault tree analysis and (b) failure modes effects criticality analysis. Both these techniques help analyze hazards and the failure modes that can lead to them.

and tools such as FTA and FMECA. ISO 14971 includes this requirement:

> The manufacturer shall ensure that those performing risk management tasks include persons with knowledge and experience appropriate to the tasks assigned to them. This shall include, where appropriate, knowledge and experience of the medical device and its use and risk management techniques.

Software quality assurance

Medical device manufacturers should establish a software quality assurance function within the R&D organization that would perform tasks commensurate with the device's risk level. SQA personnel are often responsible for a variety of software verification and validation tasks.[9]

Software verification. According to the FDA, software verification provides[3]

> … objective evidence that the design outputs of a particular phase of the software development life cycle meet all the specified requirements for that phase."

Verification looks at consistency, completeness, correctness, and documentation as the software is being

developed. Software verification's goal is to ensure that product development conforms to defined procedures and plans. Verification activities can include testing, static and dynamic analyses, simulations, and code and document inspections. All such activities should be commensurate with the device's risk and be an integral part of the software development process.

Software validation. The FDA defines software validation as[3]

> … confirmation by examination and provision of objective evidence that software specifications conform to user needs and intended uses, and that the particular requirements implemented through software can be consistently fulfilled.

Most device manufacturers perform software validation as part of the device's system validation—an activity in which SQA personnel are heavily involved. SQA personnel typically prepare a validation test plan that is based on intended use and system and software requirements. The plan leads to the creation of validation protocols that address compliance with stated requirements through device testing that reflects the device's intended use. SQA personnel thus must have some clinical expertise in addition to traditional testing skills.

Developing complex, software-based medical devices is a challenging business. Device manufacturers must understand the inherent differences between hardware and software components and establish robust software development processes that are based on recognized engineering principles appropriate for safety-critical systems. At the heart of such processes, they must incorporate risk management—from early development through product retirement. Manufacturers have a responsibility to train development and risk management teams in the use of recognized software engineering practices that promote software safety. Only then can they minimize the risk of including inherently defective software in their products. ■

References

1. US FDA Center for Devices and Radiological Health, "Medical Devices; Current Good Manufacturing Practice (CGMP) Final Rule; Quality System Regulation," 7 Oct. 1996; www.fda.gov/cdrh.
2. US FDA Center for Devices and Radiological Health, "Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices," May 2005; www.fda.gov/cdrh.
3. US FDA Center for Devices and Radiological Health, "General Principles of Software Validation, Final Guidance for Industry and FDA Staff," 2002; www.fda.gov/cdrh.
4. P.L. Jones et al., "Risk Management in the Design of Medical Device Software Systems," *Biomedical Instrumentation & Technology*, Jul./Aug. 2002, pp. 237-266.
5. B.J. Wood, "Software Risk Management for Medical Devices," *Medical Device and Diagnostic Industry*, Jan. 1999, pp. 139-156.
6. B.J. Wood and J.W. Ermes, "Applying Hazard Analysis to Medical Devices, Part 1," *Medical Device & Diagnostic Industry*, vol. 15, no. 1, 1993, pp. 79-83.
7. B.J. Wood and J.W. Ermes, "Applying Hazard Analysis to Medical Devices, Part 2," *Medical Device & Diagnostic Industry*, vol. 15, no. 3, 1993, pp. 58-64.
8. N. Leveson, *Safeware—System Safety and Computers*, Addison-Wesley, 1995.
9. S. Rakitin, *Software Verification and Validation for Practitioners and Managers*, Artech House, 2001.

Steven R. Rakitin *is president of Software Quality Consulting Inc. He helps medical device companies develop software-based devices in compliance with FDA regulations and standards. Rakitin received an MS in computer science from Rensselaer Polytechnic Institute. He is a member of the IEEE Computer Society, American Society for Quality's Biomedical and Software Divisions, and Association for the Advancement of Medical Instrumentation. Contact him at steve@swqual.com.*