Software in Medical Devices

Module 1:
• Regulations, Guidance, Standards, and Terminology
• Planning

Jeremy Jensen  Irma Sandoval-Watt
Fellow Software Quality Engineer  Senior Regulatory Strategist
Boston Scientific  Cerner Corporation
Disclosure: Boston Scientific has applications pending before FDA. Jeremy Jensen is not involved in the submission or direct support of those applications.

Disclosure: Cerner Corporation has applications pending before FDA. Irma Sandoval-Watt is involved in the submission or direct support of those applications.

Jeremy Jensen
Fellow Software Quality Engineer
Boston Scientific

Irma Sandoval-Watt
Senior Regulatory Strategist
Cerner Corporation
✓ Understand relevant FDA regulations pertaining to software
✓ Understand the key standards, technical information reports and guidance’s for software in the medical device industry
✓ Define key software terms
FDA Regulations and Guidance

PART 820 QUALITY SYSTEM REGULATION

Subpart C--Design Controls

§ 820.30 - Design controls

Subpart G--Production and Process Controls

§ 820.70 - Production and process controls

§ 820.72 - Inspection, measuring, and test equipment

§ 820.75 - Process validation

FDA General Principles of Software Validation

$20.30 - Design controls

21 CFR 820.30 (g) Design validation

21 CFR 820.70 (i) Automated Processes

Non-Medical Device Software

Medical Device Software

FDA Regulations and Guidance

AdvaMed
Advanced Medical Technology Association
## FDA Regulations

### Medical Device Software

Regulated under 21 CFR 830 – Design Controls
- Embedded (firmware)
- Accessory
- Software Only

### Non-Medical Device Software

Regulated under 21 CFR 870 – Production and Process Controls
- Software used in the design, development, and production of medical devices and software tools used to implement the quality system itself.
820.30 Design Controls

Medical Device Software
21 CFR 820.30 (g) Design Validation

Each manufacturer shall establish and maintain procedures for validating the device design. Design validation shall be performed under defined operating conditions on initial production units, lots, or batches, or their equivalents. Design validation shall ensure that devices conform to defined user needs and intended uses and shall include testing of production units under actual or simulated use conditions. **Design validation shall include software validation and risk analysis, where appropriate.** The results of the design validation, including identification of the design, method(s), the date, and the individual(s) performing the validation, shall be documented in the DHF.
820.30 Design Controls

Medical Device Software
21 CFR 820.30 (f) Design Verification

Each manufacturer shall establish and maintain procedures for verifying the device design. Design verification shall confirm that the design output meets the design input requirements. The results of the design verification, including identification of the design, method(s), the date, and the individual(s) performing the verification, shall be documented in the DHF.
When computers or automated data processing systems are used as part of production or the quality system, the manufacturer shall **validate computer software for its intended use according to an established protocol**. All software changes shall be validated before approval and issuance. These validation activities and results shall be documented.
FDA Software Guidance

**Guidance**
- Voluntary
- Define current thinking of FDA

**FDA General Principles of Software Validation (GPSV)**

Released in January 2002
Scope includes both Device Software and Non-Device Software:
- Software used in the design, development, and production of medical devices
- Software used in the design, development, production, or procurement of automated tools used for the design, development, or manufacture of medical devices
- Software tools used to implement the quality system itself
# FDA Software Guidance

<table>
<thead>
<tr>
<th>Computer Software Assurance for Manufacturing, Operations, and Quality System Software (Draft Guidance in-process)</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Proposed new FDA Guidance Document</td>
</tr>
<tr>
<td>• Will replace section 6 (VALIDATION OF AUTOMATED PROCESS EQUIPMENT AND QUALITY SYSTEM SOFTWARE) in the GPSV guidance</td>
</tr>
<tr>
<td>• Will Clarify FDA’s “current thinking” on the topic of (non-device) SW Validation</td>
</tr>
<tr>
<td>• Work started late 2016</td>
</tr>
<tr>
<td>• Sept 2018 release is planned</td>
</tr>
<tr>
<td>• Driven by FDA’s John Murray</td>
</tr>
<tr>
<td>• Siemens, Medtronic, Boston Sci, JJ and others supporting effort</td>
</tr>
<tr>
<td>• Expect new guidance on:</td>
</tr>
<tr>
<td>• Risk based testing – “Critical to Quality”</td>
</tr>
<tr>
<td>• Unscripted Testing</td>
</tr>
<tr>
<td>• Ad Hoc testing</td>
</tr>
<tr>
<td>• Leveraging the vendor validations for OTSS</td>
</tr>
<tr>
<td>• Goal of reducing level of effort for manufactures on validating Non-Device Software</td>
</tr>
</tbody>
</table>
The organization shall document procedures for the validation of the application of computer software used in the quality management system. Such software applications shall be validated prior to initial use and, as appropriate, after changes to such software or its application. The specific approach and activities associated with software validation and revalidation shall be proportionate to the risk associated with the use of the software. Records of such activities shall be maintained.
The organization shall document procedures for the validation of the application of computer software used in production and service provision ...

The organization shall document procedures for the validation of the application of computer software used for the monitoring and measurement of requirements ...
62304 Medical Device Software - Software life cycle processes

**Standards**
- Voluntary
- Can be formally recognized by the FDA
- Can result in expedited FDA submission

**62304 Medical Device Software – Software life cycle processes**

- 1st Edition release in 2006
- Adopted by the FDA and EU agencies as the standard by which they audit software used for medical devices. “State of the Art”
- Defines the life cycle requirements for medical device software. The set of processes, activities, and tasks described in this standard establishes a common framework for medical device software life cycle processes
- Outlines requirements for the following steps in the software life cycle process:
  - Software development
  - Software maintenance
  - Software risk management
  - Software configuration management
  - Software problem resolution
62304 Medical Device Software - Software life cycle processes

**Standards**
- Voluntary
- Can be formally recognized by the FDA
- Can result in expedited FDA submission

**62304 Medical Device Software – Software life cycle processes**
- Amendment 1 to the 1st Edition released in 2015
- FDA recognized in April 2016
- Not yet harmonized in the EU
- Considered “state of the art” by some notified bodies (BSI, TUV)
- Change to software safety classification
- Added requirements for legacy software
- Miscellaneous clarifications and technical changes
62304 Medical Device Software—Software life cycle processes

**Standards**
- Voluntary
- Can be formally recognized by the FDA
- Can result in expedited FDA submission

**62304 Medical Device Software – Software life cycle processes**

- 2nd Edition in process
- Work started in 2014
- The last draft did NOT pass the IEC vote in Q2 2018
- Working group dispositioning comments
- Planned release in ~2020
- Scope will be expanded from medical devices to health software
  - HEALTH SOFTWARE - software system intended to be used specifically for managing, maintaining, or improving health of individual persons, or the delivery of care.
  - Requested because internationally regulators are treating software differently
- Software Safety Class section is being modified again.
  - Severity based on injury vs Harm
- Legacy software clarifications
- Will include system security as a part of risk management
## Technical Information Reports (TIRs)

**TIRs**
- Best Practice
- Publication of the Association for the Advancement of Medical Instrumentation (AAMI)
- Not formally approved or reviewed by public
- Developed to be more responsive to underlying safety or performance issues

### TIR36:2007-Validation of Software for Regulated Processes

- Equivalent to ISO/IEC TR 80002-2
- Applicable to Non-Device Software
- Developed to assist readers in determining activities for the validation of regulated process software using a risk-based approach that applies critical thinking
- Key elements:
  - Critical thinking considering potential failures
  - “Toolbox” of techniques to increase confidence in software correctness
Technical Information Reports (TIRs)

**TIRs**
- Best Practice
- Publication of the Association for the Advancement of Medical Instrumentation (AAMI)
- Not formally approved or reviewed by public
- Developed to be more responsive to underlying safety or performance issues

**TIR45:2012-Guidance on the use of AGILE practices in the development of medical device software**
Provides recommendations for complying with international standards and U.S. Food and Drug Administration (FDA) guidance documents when using agile practices to develop medical device software

**TIR57: Principles for medical device security—Risk management**
Provides medical device manufacturers with guidance on developing a cybersecurity risk management process for their products.
List of other relevant Software Standards, TIRs and Guidance

• Changes To Medical Software Policies from 21st Century Cures Act Draft Guidance Dec 2017
• Medical Device Innovations Section of 21st Century Cures Act
• FDA Software As a Medical Device (SAMD) Clinical Evaluation Final Guidance
• FDA Final Postmarket Cybersecurity Guidance
• FDA Final Premarket Cybersecurity Guidance
• FDA Final Mobile Apps Guidance
• FDA Medical Device Data System Classification Rule
• FDA CyberSecurity for Networked Medical Devices Containing Off-the-shelf Software Guidance
• Preamble to Final FDA GPSV Guidance
• 21 CFR Part 11 Electronic Records; Electronic Signatures Rule
• 21 CFR Part 11 Feb 2003 Federal Register Notice Announcing Major Redirection for Part 11
• 21 CFR Part 11 Final Scope and Application Guidance 09/03/03
• FDA IOM 2007 Electronic Records and Computerized Complaint Data
• FDA Guidance Computerized Systems Used in Clinical Trials
• FDA Software Submission Guidance
• FDA Off-the-shelf Software Submission Guidance
List of other relevant Software Standards, TIRs and Guidance –cont.

- IEC 80001-1:2010 -- Part 1: Roles, responsibilities and activities
- IEC/TR 80001-2-1:2012 -- Part 2-1: Step by Step Risk Management of Medical IT-Networks; Practical Applications and Examples
- ISO/TR 80001-2-7:2015 -- Application guidance -- Part 2-7: Guidance for healthcare delivery organizations (HDOs) on how to self-assess their conformance with IEC 80001-1
List of other relevant Software Standards, TIRs and Guidance –cont.

- IEC/TR 80002-1:2009 Medical device software -- Part 1: Guidance on the application of ISO 14971 to medical device software
- ISO/TR 80002-2:2017 Medical device software -- Part 2: Validation of software for medical device quality systems
- IEC 82304-1:2016 Health software -- Part 1: General requirements for product safety
## FDA Terminology - Definitions

<table>
<thead>
<tr>
<th>Definition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define, Document</td>
<td>(in writing or electronically) and Implement (Do) - also known as the 3Ds</td>
</tr>
<tr>
<td>DHF (820.3(e))</td>
<td><strong>Design History File:</strong> Means a compilation of records that describes the design history of a finished device</td>
</tr>
<tr>
<td>DMR (820.3(j))</td>
<td><strong>Design Master Record:</strong> Means a compilation of records containing the procedures and specifications for a finished device. Also known as the recipe to create the finished device.</td>
</tr>
<tr>
<td>DHR (820.3(i))</td>
<td><strong>Device History Record:</strong> Specific details of a batch, lot or unit</td>
</tr>
</tbody>
</table>
# FDA Terminology - Definitions

<table>
<thead>
<tr>
<th>Design Validation (820.3(g))</th>
<th>Also known as User Validation - Confirm that the correct device was designed and meets the intended use(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Verification (820.3(f))</td>
<td>Confirm that the design outputs meet the design inputs</td>
</tr>
<tr>
<td>Software Validation (GPSV 3.1.2)</td>
<td>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.</td>
</tr>
<tr>
<td>Software Verification (GPSV 3.1.2)</td>
<td>Looks for consistency, completeness, and correctness of the software and its supporting documentation, as it is being developed</td>
</tr>
</tbody>
</table>
FDA Terminology - Definitions

Software Validation

- Planning
- Requirements
- Risk Analysis
- Evaluations
- Unit Testing
- Integration Testing
- Verification Testing
- Change Management
- Intended use / user needs
- Design Reviews
- Regression Testing
- Defect Resolution
- Traceability
Software Terminology - Objective Evidence

**Objective Evidence**
- FDA does not have a formal definition of Objective Evidence
- Often misinterpreted by industry
- Controversial with auditors

**Objective Evidence (IEC 60601-1:2005 definition 3.72)**
Information that can be proven true, based on facts obtained through observation, measurement, test, or other means
Objective Evidence

- FDA does not have a formal definition of Objective Evidence
- Often misinterpreted by industry
- Controversial with auditors

### Objective Evidence (IEC 60601-1:2005 definition 3.72)

**Objective** describes something that is based on fact and not subject to opinion; it can be observed or reviewed and doesn’t have to be interpreted by an experienced observer.

**Evidence** is something material that can be shown (i.e., provided) to others such as peers, reviewers, and regulatory officials with potential applicability to legal proceedings. Evidence could be test results, photos, and data from test instruments that are a direct output of the instrument or are transcribed by a tester. Evidence is not a feeling, hunch, or a prediction based on past experience. Evidence is material results to support an argument. In verification testing, that argument is that a test or test step passed. The evidence can be simply that the tester checks that the observed result matches the expected result. That is intuitive, objective, and when further backed up with the tester’s name and signature, meets the criteria of being evidence.

- **NOTE:** A correctly written test case does not leave the “expected result or behavior” up for interpretation.
Software Validation is Different

Software
- FDA found most defect caused by insufficient design and lack of control
- Can’t do 100% verification
- Requires Planning, User Needs, Requirements, Design, Testing at multiple level, etc.

Objectives of Software Validation
- Build Confidence in Software by:
  - Preventing Defects from getting in the Software in the first place
  - Detect any defects through Software Verification
- Ensure Software is:
  - Safe
  - Secure
  - Effective
  - Meets User’s Needs
### Regulations, Guidance, Standards and Terminology - SUMMARY

<table>
<thead>
<tr>
<th>FDA Quality System Regulations</th>
<th>Technical Information Reports</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Provides legal basis for Software Validation</td>
<td>• Developed to be more responsive to underlying safety or performance issues</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FDA Guidance</th>
<th>Terminology</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Defines current thinking of FDA</td>
<td>• Establish, DHF, DMR, DHR</td>
</tr>
<tr>
<td></td>
<td>• Design Validation, Design Verification, Software Validation, Software Verification</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Software Standards</th>
<th>Objective Evidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Claiming conformance can aid in regulatory approvals and reducing compliance risk</td>
<td>• Often misinterpreted by industry</td>
</tr>
<tr>
<td></td>
<td>• Controversial with auditors</td>
</tr>
</tbody>
</table>
PLANNING - OBJECTIVES

✓ Become familiar with Software Lifecycle tasks
✓ Identify the different paths of Traceability in software
✓ Understand the importance of Configuration Management
✓ Become aware of Software Maintenance
✓ Identify the steps for Problem Resolution
✓ Introduce example of a medical device and identify steps throughout the software lifecycle tasks
Running Example: Project “Cumulus”

We will be using a **fictional** product as an example throughout this training. “Cumulus” is a hypothetical blood pressure cuff that can be used at home and stores its readings in the cloud. Specifically:

- Product is an arm band that inflates to measure blood pressure
- It sends blood pressure reading (via Bluetooth) to the patient’s phone, where:
  - The patient can read their blood pressure from an app on the phone
  - Patient readings can be uploaded to the cloud, using the phone’s connection to the internet
  - It can send a text message to a home healthcare nurse (including patient name, location)
- We will assume this product is a medical device
SOFTWARE DEVELOPMENT PLAN

Software Lifecycle

Composed of software engineering tasks, documentation to support the software validation, specific verification and validation tasks that are appropriate for the intended use.

- Quality Planning
- System Requirements
- Software Requirements
- Software Design Specification
- Construction or Coding
- Testing
- Installation
- Operation and Support
- Maintenance
- Retirement
SOFTWARE DEVELOPMENT PLAN

Software Quality Planning

This is the ‘WHAT’ to be accomplished through the software validation effort. Defines the scope, approach, resources, schedules, types and extent of activities, tasks, work items.

Software Development Plan
• Standard, methods, tools

Software Integration and Testing Plan
• Software items (SOUP, Software Of Unknown Provenance)
• Testing Integration

Software Verification & Validation Plan
• Deliverables requiring Verification & Validation
• Required Verification & Validation tasks for each of the life cycle activities
• Milestone when deliverables are verified
• Acceptance criteria for verification of deliverables
SOFTWARE DEVELOPMENT PLAN—cont.

Software Quality Planning

This is the ‘WHAT’ to be accomplished through the software validation effort. Defines the scope, approach, resources, schedules, types and extent of activities, tasks, work items.

Software Risk Management Plan
• Activities, tasks for the risk management process

Documentation Plan
• Documents to be produced during the software development cycle

Software Configuration Management Plan
• Classes, types, categories of items to be controlled
• Software Configuration management activities and tasks
• Relationship with other support teams for development and maintenance
• Problem resolution process
A requirement can be any need or expectation for a system of its software. Requirements reflect the stated or implied needs of the customer, may be market-based, contractual, or statutory, as well as an organization’s internal requirements.

- All Software Inputs & Outputs
- All functional and capability requirements that the software system will perform
- Software-driven alarms, warnings, and operator message
- Security
- User interfaces, how users will interact with system
- Interfaces software system and other systems
- Data definition and database requirements
Software System Requirements

All requirements to be evaluated for:
• Accuracy
• Completeness
• Correctness
• Clarity
• Testability

• What constitutes error and how errors should be handled
• Installation and acceptance requirements
• Methods of Operation and Maintenance
• IT network
• User maintenance
• Regulatory
• Risk Control measures
DESIGN

Software Design

Design is the process where the software requirements specifications are translated into logical and physical representation of the software to be implemented. Description of what the software should do and how it should do it.

- Software requirements specifications
- Software risk analysis
- Development procedures and coding guidelines
- System documentation (system context, relationship of hardware, software, and the physical environment)
- Parameters to be recorded and measured
DESIGN – cont.

Software Design

Acceptance criteria:
• Proper event sequence
• Data and control flow
• Planned resource allocation
• Fault handling (error definition, isolation, and recovery)
• Self-diagnostics
• Memory management and memory overflows
• Boundary conditions

- Logical structure (control logic), logical processing steps (algorithms)
- Data structures and flow diagrams
- Variables definitions and where used
- Error, alarm, warning messages
- Supporting software (drivers, operating systems)
- Security measures (physical and logical)
Software may be constructed either by coding (programming) or by assembling together previously coded software components (code libraries, off-the-shelf software) for use in new application.

- Traceability Analyses
  - Source Code to Design Specification (and vice versa)
  - Test Cases to Source Code and to Design Specification
- Source Code and Source Code Documentation Evaluation
- Test Procedure and Test Case Generation (module, integration, system, and acceptance)
Source Code Traceability

Source Code Traceability analysis is an important tool to verify that all code is linked to established specifications and established test procedures.

The source code traceability analysis should conducted and documented to verify the following:

• Each element has been implemented
• Modules and functions can be traced back
• Tests for modules and functions can be traced back
• Test for modules and functions can be traced to source code for the same modules and functions
Software testing entails running software products under known conditions with defined inputs and documented outcomes that can be compared to their predefined expectations (criteria).
White-box Testing

This type of testing is also known as code-based, or structural testing.

These test cases challenge the control decisions made by the program.
Black-box Testing

This type of testing is called definition-based or specification-based.

These test cases challenge the intended use or functionality of a program, and the program’s internal and external interfaces.
Integration Testing

In addition to structural and specification-based testing, integration testing focuses on multiple layers and varies based on the system where the device is being integrated with (External Interfaces).
System Testing

This type of testing demonstrates that all specified functionality exists and that the software product is trustworthy. System level software testing addresses functional concerns and the following elements:

- Performance issues
- Response to stress conditions
- Operation of internal and external security features
- Effectiveness of recovery procedures, including disaster recovery
- Usability
- Compatibility with other software products
- Behavior in each of the defined hardware configurations
- Accuracy of documentation
User Testing

This type of testing becomes essential for software validation because, as part of the Quality System Regulation, installation and inspection procedures, documentation, and testing to demonstrate proper installation are required.

This testing is also known as:

- Beta Testing
- Site Validation
- User Acceptance Test
- Installation Verification
- Installation Testing
Software Changes

Once a software product has been baselined (approved), any change should have its own “mini life cycle” testing.

Changes can be the result of:

• Debugging that finds an error and it is corrected

• New or changed requirements, also known as “requirements creep”

• Modified designs as more effective or efficient implementations are found
Regression Testing

Regression analysis and testing are employed to provide assurance that a change has not created problems elsewhere in the software product.
Release of Software

This is when all software is verified to be complete and the results have been evaluated before software release.

Ensure that:

• Known anomalies are documented
• Residual know anomalies do not contribute to unacceptable risk
• Version
• All task are complete
• Archive software
• Delivery
Steps:

- Identify software hazardous situations
- Identify potential causes of contribution to hazardous situations
  - Incorrect or incomplete specs of functionality
  - Software defects
  - Failure of unexpected results from SOUP
  - Hardware failures
  - Reasonable foreseeable misuse
- Evaluate published SOUP anomaly lists
- Document potential causes
Configuration Management

This is the process for managing modifications to existing software systems.

Steps:

• Perform risk control measures
  ✓ Define risk control measures
  ✓ Risk control measures implemented in software

• Verify risk control measures
  ✓ Document traceability

• Perform risk management of software changes
  ✓ Analyze changes with respect to safety
  ✓ Analyze impact of changes on risk control measures
  ✓ Perform risk management activities
# MAINTENANCE

## Hardware

Includes:
- Corrective changes
- Preventive actions,
- Component replacement

## Software

Includes:
- Corrective,
- Perfective, and
- Adaptive maintenance but does not include preventive actions or component replacement

---

Difference between Hardware and Software is their failure/error mechanisms
The maintenance plan should address:

- Procedure for:
  - Receiving
  - Documenting
  - Evaluating
  - Resolving
  - Tracking

- Criteria for determining if feedback is a problem

- Use Software Risk Management Process

- Use of the Software Problem Resolution Process

- Use software configuration management process

- Procedures to evaluate
  - Upgrades
  - Bug fixes
  - Patches
  - Obsolescence of SOUP
Problem Reports

Problem reports shall be prepared for each problem detected in the device software. The reports will need to be identified by criticality:

- Effect on performance,
- Safety, or
- Security.

Steps:

- Investigate the problem
- Advise relevant parties
- Use change control process
- Maintain records
- Analyze problems for trends
- Verify software problem resolution
- Test documentation contents
Reference Documents

• IEC 62304: 2006 Edition 1.1. 2015-06, Medical device software – Software life cycle processes

• Food Drug Administration Center for Devices and Radiological Health and Center for Biologics Evaluation and Research, General Principles of Software Validation; Final Guidance for Industry and FDA Staff, Issued January 11, 2002
QUESTIONS?