Medical Device Software - Software Life Cycle Processes

IEC 62304

The CDRH Software Education Program
Center for Devices and Radiological Health
US Food & Drug Administration

Credits

- John F. Murray
  Software Compliance Expert
  U.S. Food and Drug Administration
- Marcie R. Williams
  Medical Device Fellow
  Ph.D. Candidate, Georgia Institute of Technology
- IEC 62304 Working Group

History of IEC 62304

- Good Manufacturing Practices – 1976
- Quality Systems Regulation – 1996
  - (Design Controls)
- General Principles of Software Validation – 1998-2002
- SW68 – 2001
- IEC 62304 - 2006

There is no known method to guarantee 100 % SAFETY for any kind of software.

(Annex B.4)

Software Assurance

- Establishing the safety and effectiveness of medical device software

(Introduction ¶ 1)

- Method:
  - – Define the intended use of the software
  - – Demonstrate that the software fulfills those intentions
  - – Demonstrate that the software does not cause any unacceptable risks

(Introduction ¶ 1)

Purpose of IEC 62304

- To define the life cycle requirements for medical device software

  (Introduction ¶ 2)

- To establish a common framework for medical device software life cycle processes
  - Life cycle should be well described and broken into processes, activities, and tasks which will be performed
  - Testing is not sufficient to establish safety

  (1 Scope, 1.1 & Annex A.1)

Field of Application

- Development and Maintenance of Medical Device Software
  (1 Scope, 1.2)
- Medical Device Software =
  - Software which is a medical device
  - Software which is part of a medical device

(1 Scope, 1.2)

Compliance

- Quality Management System
  - ISO 13485

  (4 General Requirements, 4.1)

- Risk Management Process
  - ISO 14971

  (4 General Requirements, 4.2)

- Implement the processes, activities, and tasks described in this standard (IEC 62304)
  - No specific organizational structure for the manufacturer is specified

  (1 Scope, 1.4)

General Requirements

- Documentation of tasks shall be produced
  - No specific format for this documentation is specified

(Introduction ¶ 7)

- A life cycle shall be established
  - Map processes, activities, and tasks in this standard to the life cycle model of the manufacturer’s choosing
  - No particular life cycle is specified

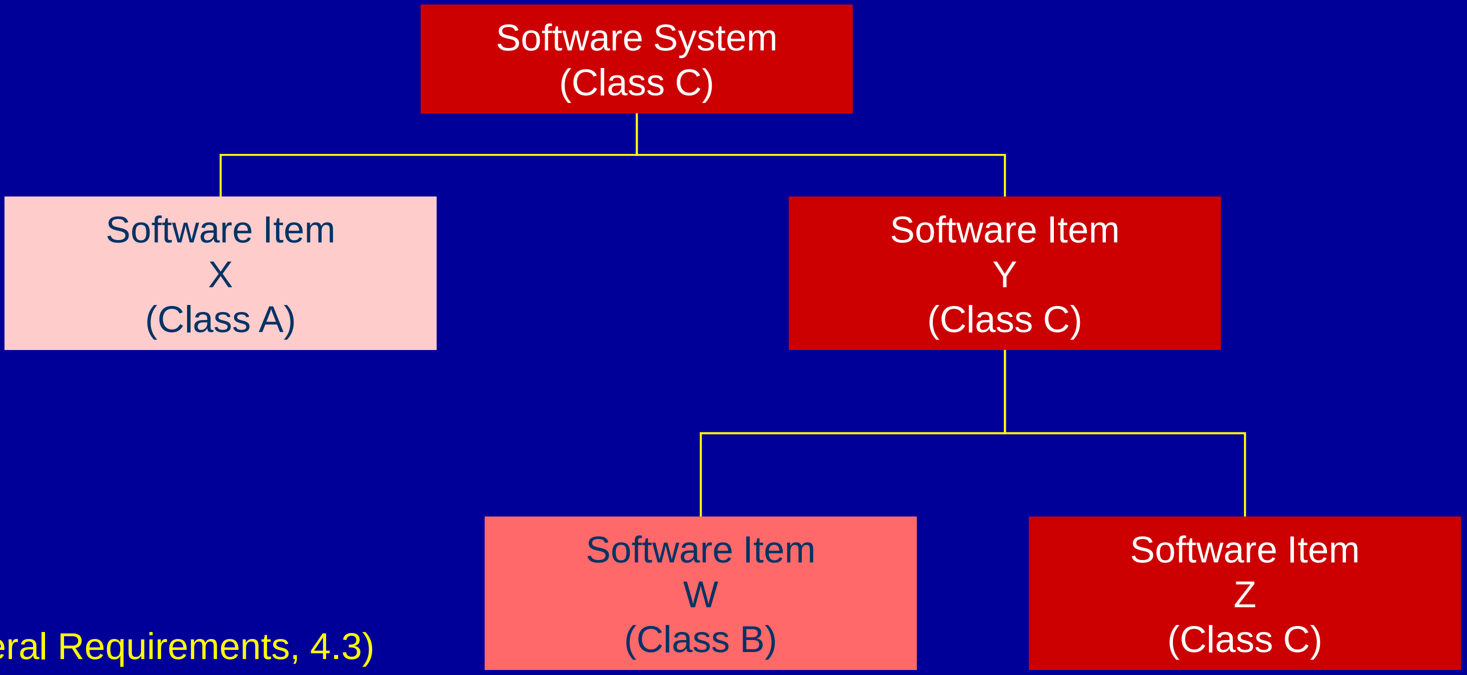
(Introduction ¶ 8)

Classification Schemes

Software Safety Classification **IEC 62304** (4 General Requirements, 4.3)	**Level of Concern** **Guidance for the Content of Pre-market Submissions for Software Contained in Medical Devices**
Class A: No injury or damage to health is possible	Minor: Failures or latent design flaws are unlikely to cause any injury
Class B: Non-Serious injury is possible	Moderate: Failure or latent design flaw could directly or indirectly result in minor injury
Class C: Death or Serious injury is possible	Major: Failure or flaw could directly or indirectly result in death or serious injury

Software Safety Classification

- Risk Control
- Segregation of Software



(4 General Requirements, 4.3)

Benefits of IEC 62304

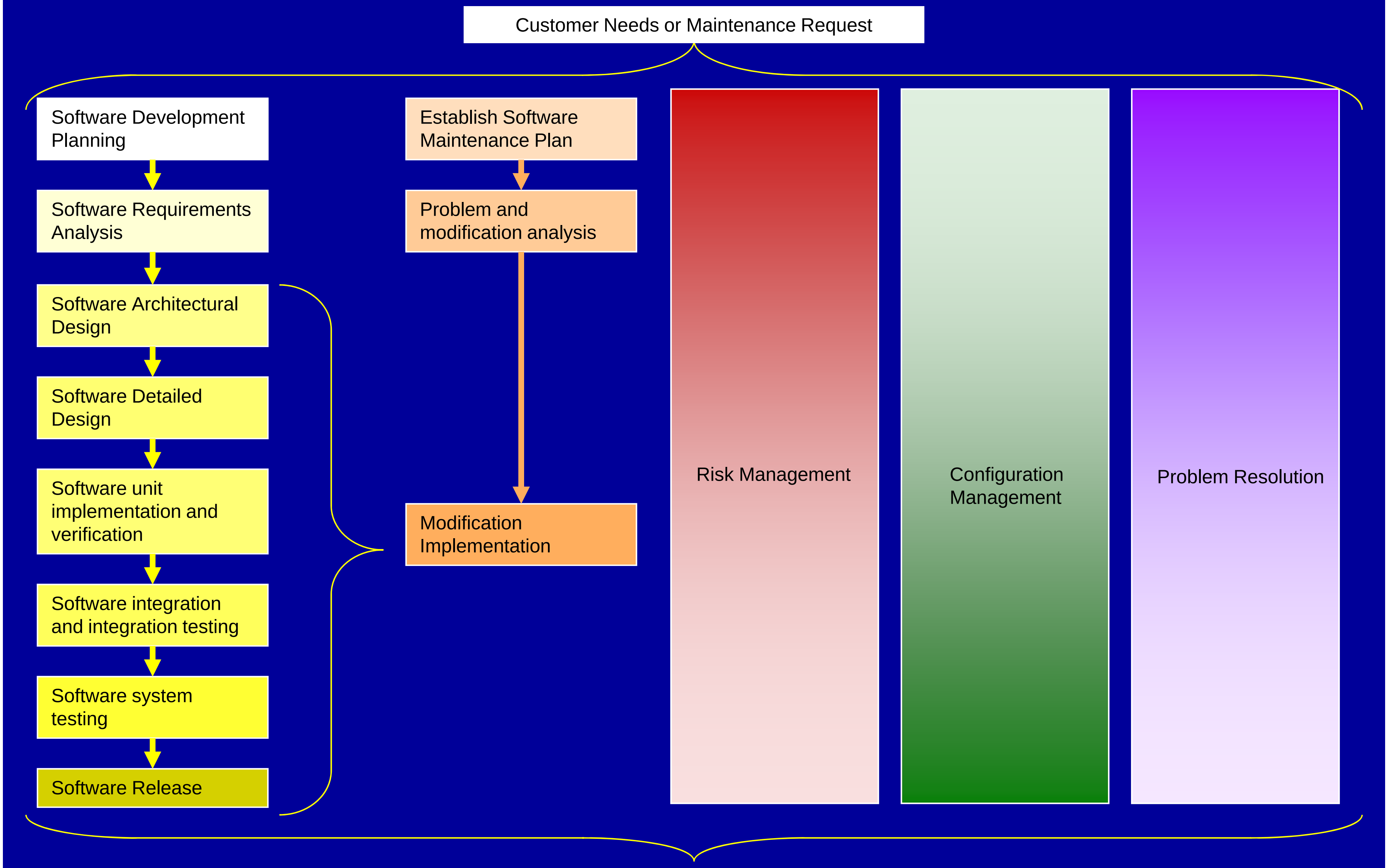
- Enhances the reliability of the software by requiring detail or rigor in the design, testing, or verification

  (Annex A.1)

- Enhances the safety of medical device software

Life Cycle Processes

- Software Development Process
- Software Risk Management Process
- Software Configuration Process
- Software Problem Resolution Process
- Software Maintenance Process



Customer Needs or Maintenance Request

Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

Establish Software Maintenance Plan

Problem and modification analysis

Modification Implementation

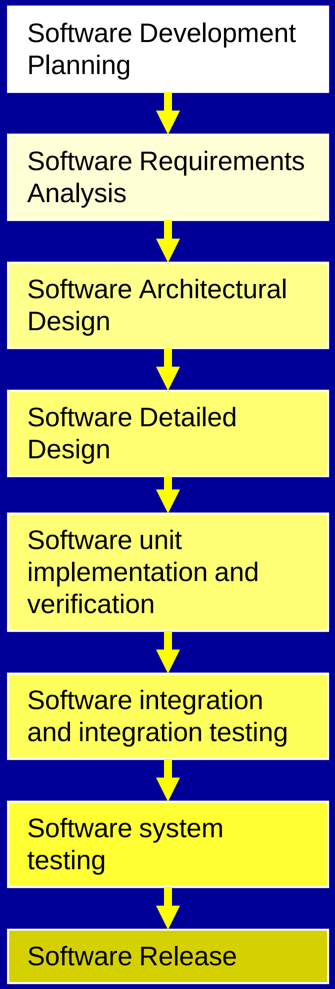
Risk Management

Configuration Management

Problem Resolution

Customer Needs and Maintenance Requests Satisfied

(Introduction, Figures 1 & 2)



Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

Software Development Process

5.1 Software Development Planning

What is Software Development Planning?

- Thinking through the software development process and creating a document which describes all of the events that will occur during the software life cycle
  - Planning performed before you DO the work
  - Allows for allocation of time and resources

Planning is an iterative activity that should be re-examined and updated as development progresses.

(Annex B.5.1)

Software Development Plan

- Manufacturer shall establish a plan
- Plan should be appropriate to the scope, magnitude, and software safety classifications of the system to be developed
- Documentation of tasks to be performed may be in a single plan or multiple plans
  - May also reference previously existing policies and procedures for the manufacturer

(5 Software Development Process, 5.1.1 and Annex B.5.1)

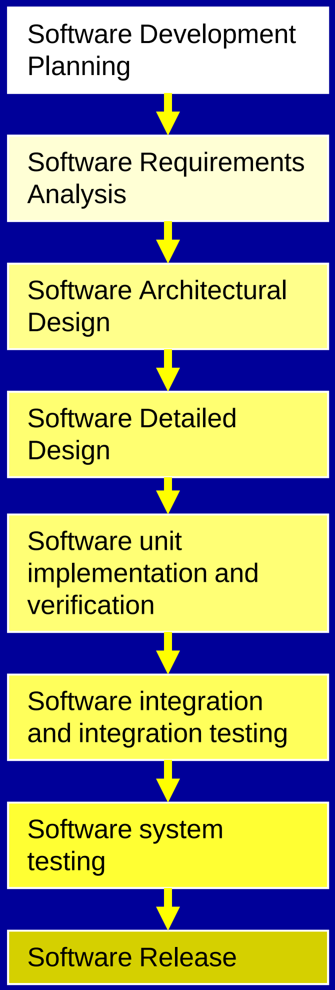
System Engineering vs. Software Engineering

- Software requirements shall reference system requirements
- Plan should coordinate software development with a quality management system

(5 Software Development Process, 5.1.3)

Types of Planning

- Software Integration Planning
- Software Verification Planning
- Risk Management Planning
- Documentation Planning
- Configuration Management Planning



Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

5.2 Software Requirements Analysis

What is Software Requirements Analysis?

- Establishing and verifying software requirements
- Software requirements are:
  - Formally documented specifications of what the software does to meet the customer needs
- System and software requirements might be the same if the software is a software-only device

(Annex B.5.2)

Value of Software Requirement Analysis

- Establishing verifiable requirements is essential for:
  - Determining what is to be built
  - Determining that the software exhibits acceptable behavior
  - Demonstrating that the software is ready for use

(Annex B.5.2)

Software Requirements

- Content:
  - Functional and capability requirements
  - Software system inputs and outputs
  - Interfaces between the software system and other systems
  - Software-driven alarms, warnings, and operator messages
  - Security requirements
  - Usability engineering requirements sensitive to human errors and training

(5 Software Development Process, 5.2.2)

Software Requirements

- Content, continued:
  - Data definition and database requirements
  - Installation and acceptance requirements
  - Requirements related to methods of operation and maintenance
  - User documentation to be developed
  - User maintenance requirements
  - Regulatory requirements

(5 Software Development Process, 5.2.2)

Risk Control & Software Requirements

- Requirements should include risk control measures

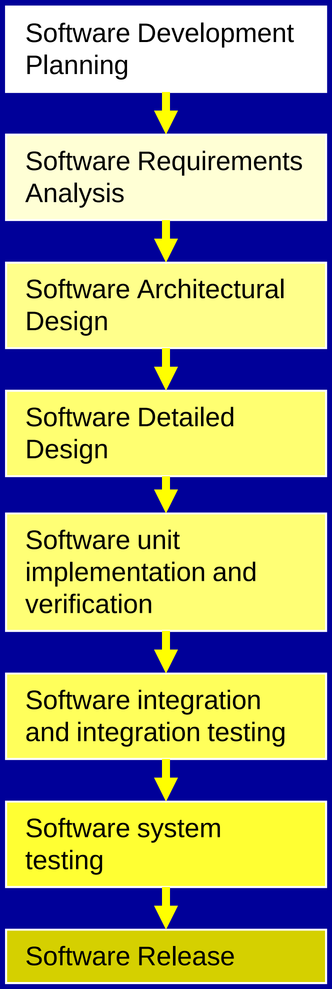
- When software requirements are established, risk analysis should be re-evaluated and kept updated

(5 Software Development Process, 5.2.3 & 5.2.4)

Characteristics of Good Software Requirements

- Implement system requirements (including risk control)
- Are traceable to system requirements
- Can be uniquely identified
- Do not contradict each other
- Language is not ambiguous
- Permit establishment of test criteria
- Permit performance of tests to evaluate if test criteria have been met

(5 Software Development Process, 5.2.6 and Annex B.5.2)



Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

5.3 Software Architectural Design

Software Architectural Design

- Architecture describes software structure and identifies software items
- Describes interfaces for software items
- Identifies segregation necessary for risk control

(5 Software Development Process, 5.3.1-5.3.5)

Architectural Design and Off-the-Shelf Software

- Specifies functional and performance requirements of off-the-shelf software
- Specifies hardware and software required by off-the-shelf software

(5 Software Development Process, 5.3.1-5.3.6)

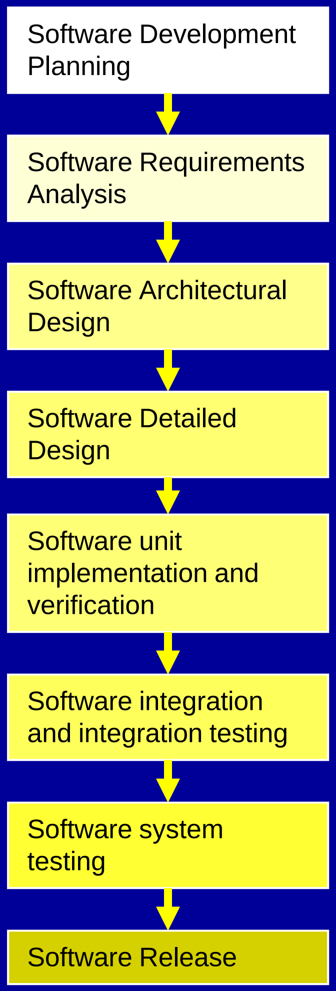
Software Architecture Verification

- Verify and Document that:
  - Architecture implements system and software requirements, including risk control
  - Architecture supports interfaces between software and hardware
  - Architecture supports proper operation of off-the-shelf software

(5 Software Development Process, 5.3.6)

Value of Architectural Design

- Risk Management
- Allocation of Resources
- Problem Definition



Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

5.4 Software Detailed Design

What is Detailed Design?

- Refining software items described in the architecture to create software units and interfaces
- Each software unit can be tested separately
- The software design fills in the details necessary to construct the software
  - Programmers should not be required to make ad hoc decisions during coding

(Annex B.5.4)

Detailed Design

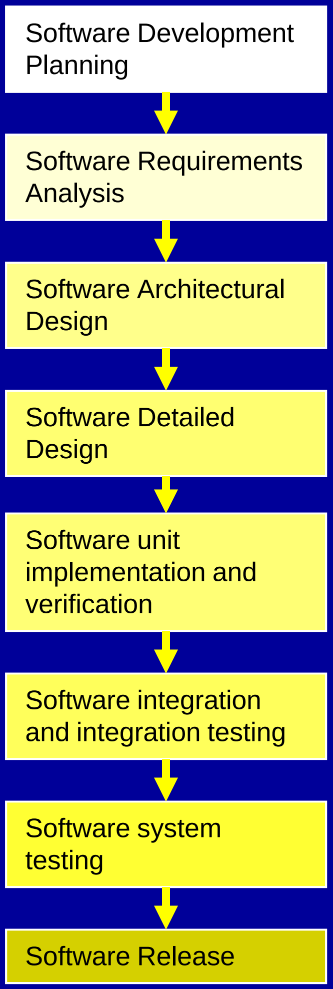
- Develop detailed design for each software unit
- Develop detailed design for interfaces
- Verify and document that the software unit:
  - Implements the architectural design
  - Is free from contradiction with the architecture

(5 Software Development Process, 5.4.1-5.4.4)

Value of Detailed Design

- Form of design control
  - Allows for review and management oversight
- Minimizes defect insertion
- If the detailed design contains defects, the code will not implement the requirements correctly

(Annex B.5.4)



Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

5.5 Software Unit Implementation and Verification

What is Unit Implementation and Verification?

- Translating the detailed design into source code
- This is the point where decomposition of the specifications ends and composition of the executable software begins.
- To consistently achieve desired results, coding standards should be used.
- The source code should be verified.

(Annex B.5.5)

Implementation and Verification

- Implement each software unit
  - Unit should have a configuration ID
- Verify each software unit according to procedures established by the manufacturer

(5 Software Development Process, 5.5.1 & 5.5.2)

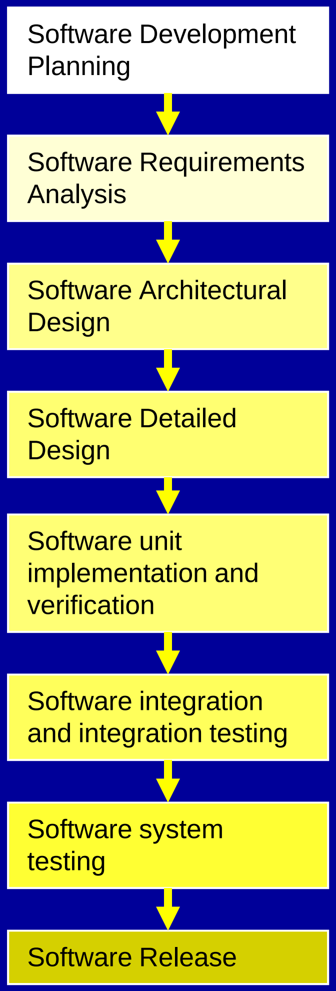
Acceptance Criteria

- Manufacturer must establish acceptance criteria for each software unit
- As appropriate, criteria should address:
  - Software requirements
  - Conformance with programming procedures or coding standards
  - Event sequence
  - Data and control flow
  - Resource allocation
  - Fault handling
  - Initialization of variables
  - Self diagnostics
  - Memory management
  - Boundary Conditions

(5 Software Development Process, 5.5.3 & 5.5.4)

Value of Unit Implementation

- The medical device software should perform as intended if the code correctly implements a properly developed detailed design



Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

5.6 Software Integration and Integration Testing

What is software integration and testing?

- Combining software units to form aggregate software items
- Combining software items into higher aggregated software items
- Verify that the resulting software items behave as intended

(Annex B.5.6)

Integration

- Integrate software units according to integration plan
- Test integrated software according to integration plan
- Evaluate test results and procedures for correctness
- Perform regression tests on previously integrated software as appropriate

(5 Software Development Process, 5.6.1-5.6.5)

Integration Testing

- Focus on transfer of data and control across a software item’s internal and external interfaces
- Rigor of testing and level of detail commensurate with:
  - the risk associated with the device
  - the device’s dependence on software for potentially hazardous functions
  - the role of specific software items in higher risk functions
- Items that have an effect on safety should be subject to more direct, thorough, and detailed tests.

(Annex B.5.6)

Types of Testing -The Toolbox-

- White Box Testing
  - Glass Box
  - Structural
  - Clear Box
  - Open Box
- Black Box Testing
  - Behavioral
  - Functional
  - Opaque-box
  - Closed-box

(Annex B.5.6)

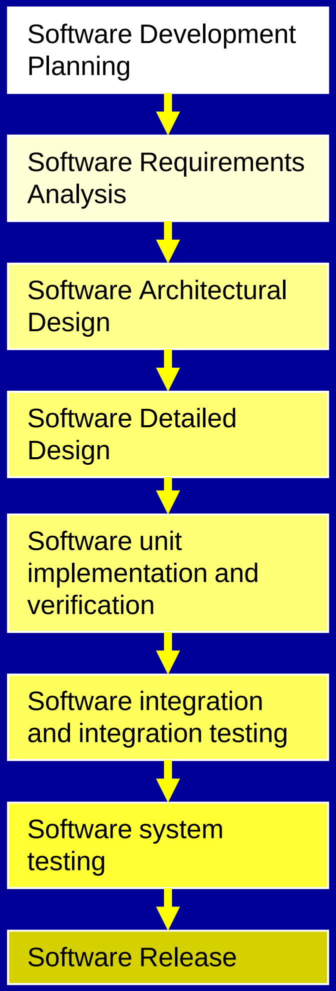
Integration Records and Problem Resolution

- Integration records should include:
  - Test results and a list of anomalies
  - Information to permit a repeat of the test
  - Identification of tester
- Problem Resolution
  - Anomalies shall be entered into the software problem resolution process

(5 Software Development Process, 5.6.7-5.6.8)

Value of Integration

- Verifies that the software behaves as intended
- Verifies that transfer of data and control across interfaces performs correctly
- Provides assurance commensurate with the risk of the device’s dependence on software



Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

5.7 Software System Testing

What is Software System Testing?

- Performing tests and verification procedures on the entire software system following integration

System Testing

- Establish and perform tests on software system
- Enter anomalies into software problem resolution process
- Retest if changes are made
- Verify that:
  - Verification methods and test procedures are appropriate
  - System test procedures trace to software requirements
  - All software requirements have been tested or verified
  - Test results meet the require pass/fail criteria

(5 Software Development Process, 5.7.1 – 5.7.4)

Planning for System Testing

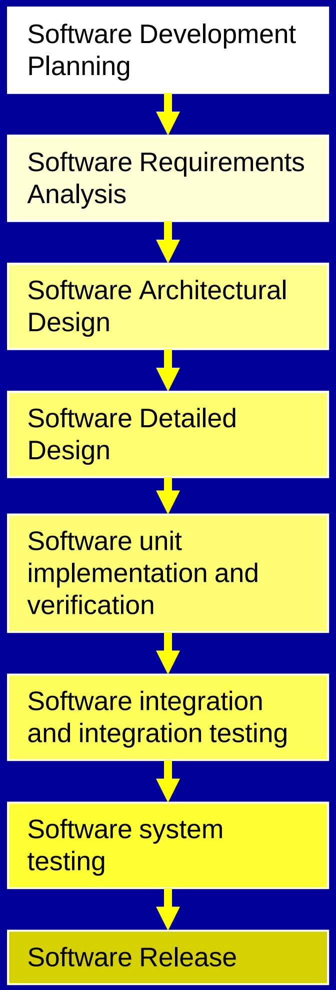
- Software and hardware tests can be performed in a simulated or actual environment
- Test responsibilities can be dispersed across various locations and organizations
  - It is ultimately the manufacturers responsibility to ensure that the software functions properly for its intended use
- Anomalies that are identified should be evaluated for their effect on the safety of the device
  - If it is decided that these anomalies will not be fixed a rationale for this must be documented

(Annex B.5.7)

Value of System Testing

- Testing (attempts to) demonstrate that the specified functionality exists by verifying that the requirements for the software have been successfully implemented.
- Results in a Finished Device

(Annex B.5.7)



Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

5.8 Software Release

Prior to Software Release

- Ensure verification is complete
- Document known residual anomalies
- Evaluate known residual anomalies
- Document released versions
- Document how software was created
- Ensure activities and tasks in design plan are complete\
- Archive software
- Assure repeatability of software release

(5 Software Development Process, 5.8.1-5.8.8)

Value of Software Release Controls

- Ensures that the manufacturer documents the version of the medical device being released
- Allows manufacturer to demonstrate that the software was developed using a quality system
- Allows manufacturer to retrieve the software and the tools used for its generation in case it is needed for future use
- Provides documentation for the device master record and the device history record (820.181 & 820.184)

(Annex B.5.8)

GPSV	IEC 62304
5.2.1 Quality Planning	5.1 Software development planning
5.2.2 Requirements	5.2 Software requirements analysis
5.2.3 Design	5.3 Software architectural design 5.4 Software detailed design
5.2.4 Construction or Coding	5.5 Software unit implementation and verification 5.6 Software integration and integration testing
5.2.5 Testing by the software developer	5.5 Software unit implementation and verification 5.6 Software integration and integration testing 5.7 Software system testing
5.2.6 User Site Testing	5.7 Software system testing
5.2.7 Maintenance and Software Changes	6 Software Maintenance Process

Where's Waldo?

- Software Development Process
- Software Risk Management Process
- Software Configuration Process
- Software Problem Resolution Process
- Software Maintenance Process

Risk Management

Software Risk Management Process

Important Concepts for Risk Management

- Software risk management is a part of overall medical device risk management
  - Cannot be adequately addressed in isolation
- Risk Management process in this standard provides additional risk control requirements specifically for software
- This process is included because:
  - Manufacturers and regulators need to understand the minimum risk control measures necessary in their area of responsibility (software)
  - The general risk management standard (ISO 14971) does not specifically address the risk control of software and its place in the software development life cycle

(Annex B.7.1)

Requirements of Risk Management Process

- Use of a process that is compliant with ISO 14971
- Must have a documented software risk management plan
- Hazard analysis must identify hazardous situations and risk control measures to reduce the probability and/or the severity of these situations to an acceptable level
- Risk control measures will be assigned to software functions that are expected to implement those risk control measures

(Annex B.7.1)

Risk Management

7.1 Software and Hazardous Situations

7.1 Software and Hazardous Situations

- Identify software items that contribute to a hazardous situation
- Identify potential causes of this hazard

(7 Software Risk Management Process, 7.1.1 & 7.1.2)

7.1 Software and Hazardous Situations

- Evaluate Published SOUP anomalies list
  - If SOUP is a potential cause of a hazardous situation
  - Identify any sequence of events that could lead to such a situation

(7 Software Risk Management Process, 7.1.3)

7.1 Software and Hazardous Situations

- Document:
  - – Potential causes of the software item contributing to a hazardous situation
  - – Sequences of events that could result in a hazardous situation

(7 Software Risk Management Process, 7.1.4 - 7.1.5)

Risk Management

7.2 Risk Control Measures

Define Risk Control Measures

For each potential cause of the software item contributing to a hazardous situation documented in the risk management file, the manufacturer shall define and document risk control measures.

(7 Software Risk Management Process, 7.2.1)

Implement Risk Control Measures

- Manufacturer is required to:
  - Include the risk control measure in the software requirements
  - Assign a software safety class to the software item based on the possible effects of the hazard that the risk control measure is controlling
  - Develop the software item in accordance with the software development process

(7 Software Risk Management Process, 7.2.2)

Risk Management

7.3 Verification of Risk Control Measures

Verification

- Each risk control measure must be documented and verified
  - Verification must also be documented
- The manufacturer shall evaluate risk control measures to identify any new sequences of events that could result in a hazardous situation

(7 Software Risk Management Process, 7.3.1 & 7.3.2)

Traceability

- Document traceability :
  - From the hazardous situation to the software item
  - From the software item to the specific software cause
  - From the software cause to the risk control measure
  - From the risk control measure to verification of the risk control measure

(7 Software Risk Management Process, 7.3.3)

Risk Management

7.4 Risk Management of Software Changes

7.4 Risk Management of Software Changes

- Analyze changes with respect to safety
- Analyze the impact of changes on risk control measures
- Perform risk management activities based on this analysis

(7 Software Risk Management Process, 7.4)

Value of Risk Management Process

- Method used to identify items of medical device software associated with hazards
- Method used to identify hazards that need software as a risk control measure
- Method used to determine allocation of resources and the appropriate critical parts of software

Configuration Management

Software Configuration Management Process

What is Software Configuration Management?

A process of applying administrative and technical procedures throughout the software life cycle to identify and define software items, including documentation

(Annex B.8)

8.1 Configuration Identification

- Establish a scheme to identify configuration items
- Configuration items should include SOUP
- Document configuration items and their versions within the software system

(8 Software Risk Management Process, 8.1)

8.2 Change Control

- Approve Change Requests
- Implement Changes
- Verify Changes

(8 Software Risk Management Process, 8.2.1 – 8.2.3)

8.2 Provide Means for Traceability

- Audit trail for:
  - Change requests
  - Problem reports
  - Approval of change requests

(8 Software Risk Management Process, 8.2.4)

Value of Software Configuration Management

- Necessary to recreate a software item
- Necessary to identify the constituent parts of a software item
- Provides a history of the changes that have been made to a software item

(Annex B.8)

Problem Resolution

Software Problem Resolution Process

What is software problem resolution?

- A process for analyzing and resolving problems, whatever their nature or source.
  - This includes those problems discovered during the execution of development, maintenance, or other processes.

(Annex B.9)

Prepare Problem Reports

- Problem reports should be classified according to:
  - Type
  - Scope
  - Criticality

(9 Software Risk Management Process, 9.1)

Investigate the Problem

- The manufacturer shall:
  - Investigate the problem and identify the causes
  - Evaluate the problem’s relevance to safety (using Risk Management Process)
  - Document the outcome of the investigation and evaluation
  - Create a change request as needed or document rationale for taking no action

(9 Software Risk Management Process, 9.2)

Advise, Maintain, and Analyze

- Advise relevant parties of the problem
- Maintain records of problem reports and their resolution
- Analyze problems for trends

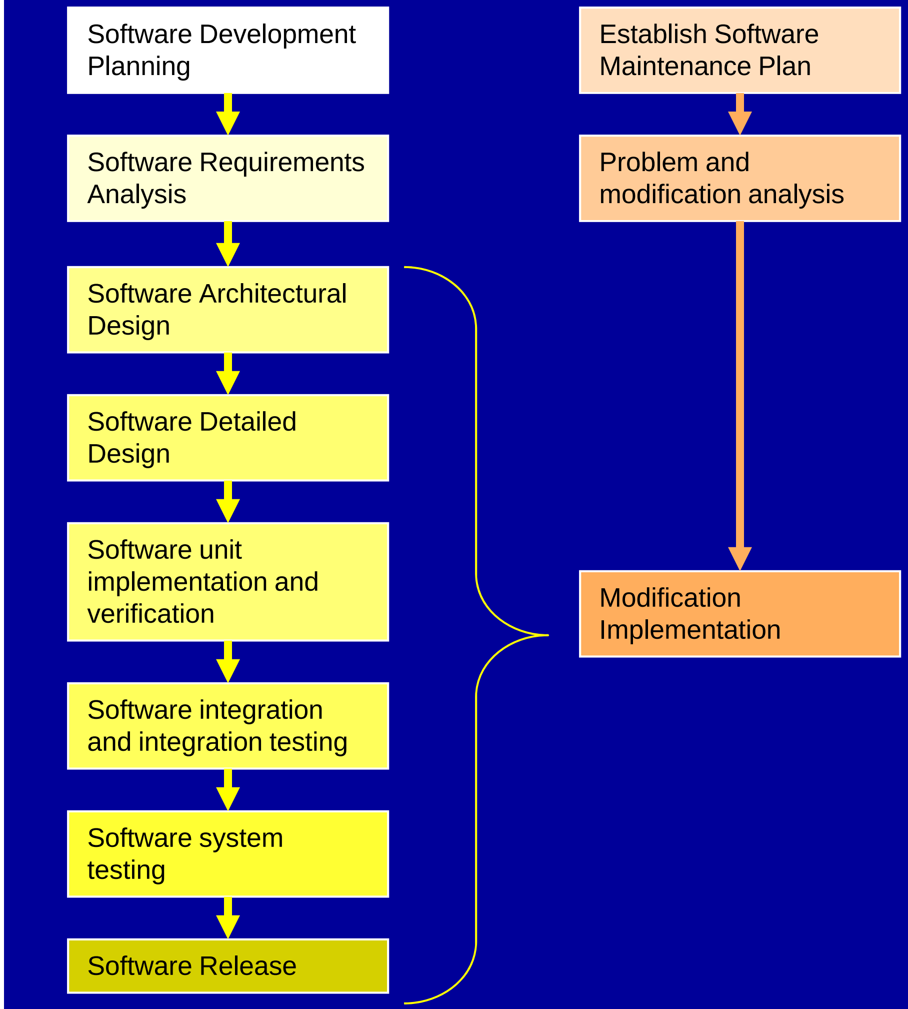
(9 Software Risk Management Process, 9.3 – 9.6)

Value of Problem Resolution Process

- Ensures that discovered problems are analyzed and evaluated for possible relevance to safety
- Ensures that problems are handled in a way which conforms with quality systems and risk management processes

(Annex B.9)

Software Maintenance Process


Software Development Planning
Software Requirements Analysis
Software Architectural Design
Software Detailed Design
Software unit implementation and verification
Software integration and integration testing
Software system testing
Software Release
Establish Software Maintenance Plan
Problem and modification analysis
Modification Implementation

Maintenance Process

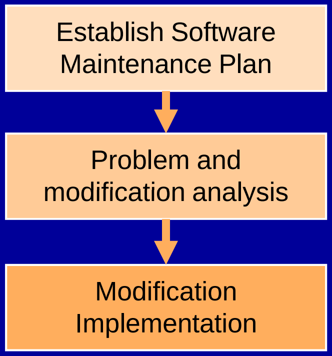
1. Establish Plan
2. Problems and Modification Analysis
3. Implement Changes

(6 Software Maintenance Process)

Maintenance Process vs. Software Development Process

- Manufacturer may use a smaller process than the full software development process to implement rapid changes to urgent problems
- The manufacturer not only addresses the problem but also satisfies local regulations

(Annex B.6.1)



Establish Software Maintenance Plan

Problem and modification analysis

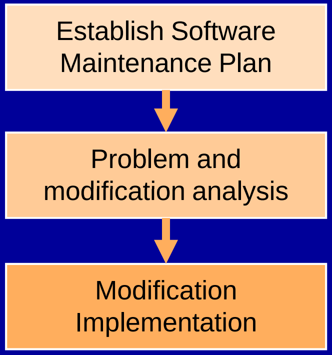
Modification Implementation

6.1 Software Maintenance Plan

Maintenance Plan

- Should address the following
  - Procedures for receiving, documenting, evaluating, resolving, and tracking feedback after release of the medical device software
  - Criteria for whether feedback is considered a problem
  - Use of the risk management process
  - Use of the problem resolution process
  - Use of the configuration management process
  - Procedure to evaluate and implement upgrades, bug fixes, patches, and obsolescence in off-the-shelf software

(6 Software Maintenance Process, 6.1)



Establish Software Maintenance Plan

Problem and modification analysis

Modification Implementation

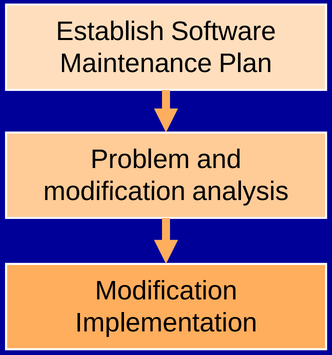
6.2 Problem and Modification Analysis

Change Requests

- Evaluate and approve change requests which modify released software products

- Inform users and regulators about
  - Problems in release software and the consequences of continued unchanged use
  - Available changes to the software and how to obtain and install the changes

(6 Software Maintenance Process, 6.2.4 & 6.2.5)



Establish Software Maintenance Plan

Problem and modification analysis

Modification Implementation

6.3 Modification Implementation

Modification Implementation

- Use software development process to implement modifications
- Re-release modified software according to software release plans (5.8)

(6 Software Maintenance Process, 6.3.1 & 6.3.2)

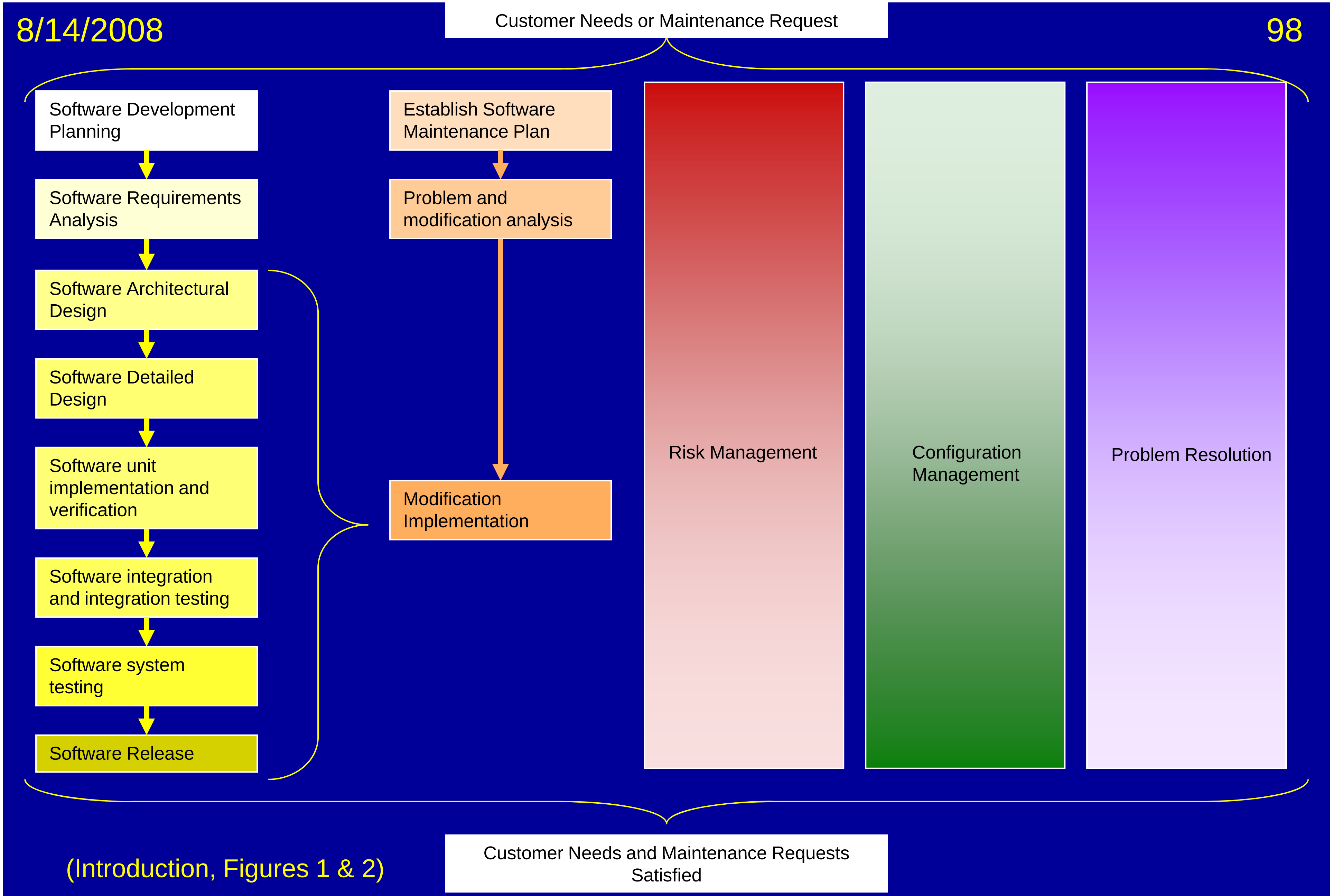
Maintenance and Problem Resolution Actions

- Safety related problem reports are addressed and reported to regulatory authorities and users
- Software products are re-validated and re-released after modification
- The manufacturer considers what other products might be affected and takes appropriate action
- Analyses problem reports and identifies all implications of the problem
- Decides on a number of changes and identifies all their side-effects
- Implements the changes while maintaining consistency with configuration management and risk management
- Verifies the implementation of the changes

(Annex B.6.2)

Value of Software Maintenance Process

- Software is always changing
- A smaller process for maintenance can be used than the full software development process
- Process allows the manufacturer to modify released software while preserving its integrity



Customer Needs or Maintenance Request

Software Development Planning

Software Requirements Analysis

Software Architectural Design

Software Detailed Design

Software unit implementation and verification

Software integration and integration testing

Software system testing

Software Release

Establish Software Maintenance Plan

Problem and modification analysis

Modification Implementation

Risk Management

Configuration Management

Problem Resolution

Customer Needs and Maintenance Requests Satisfied

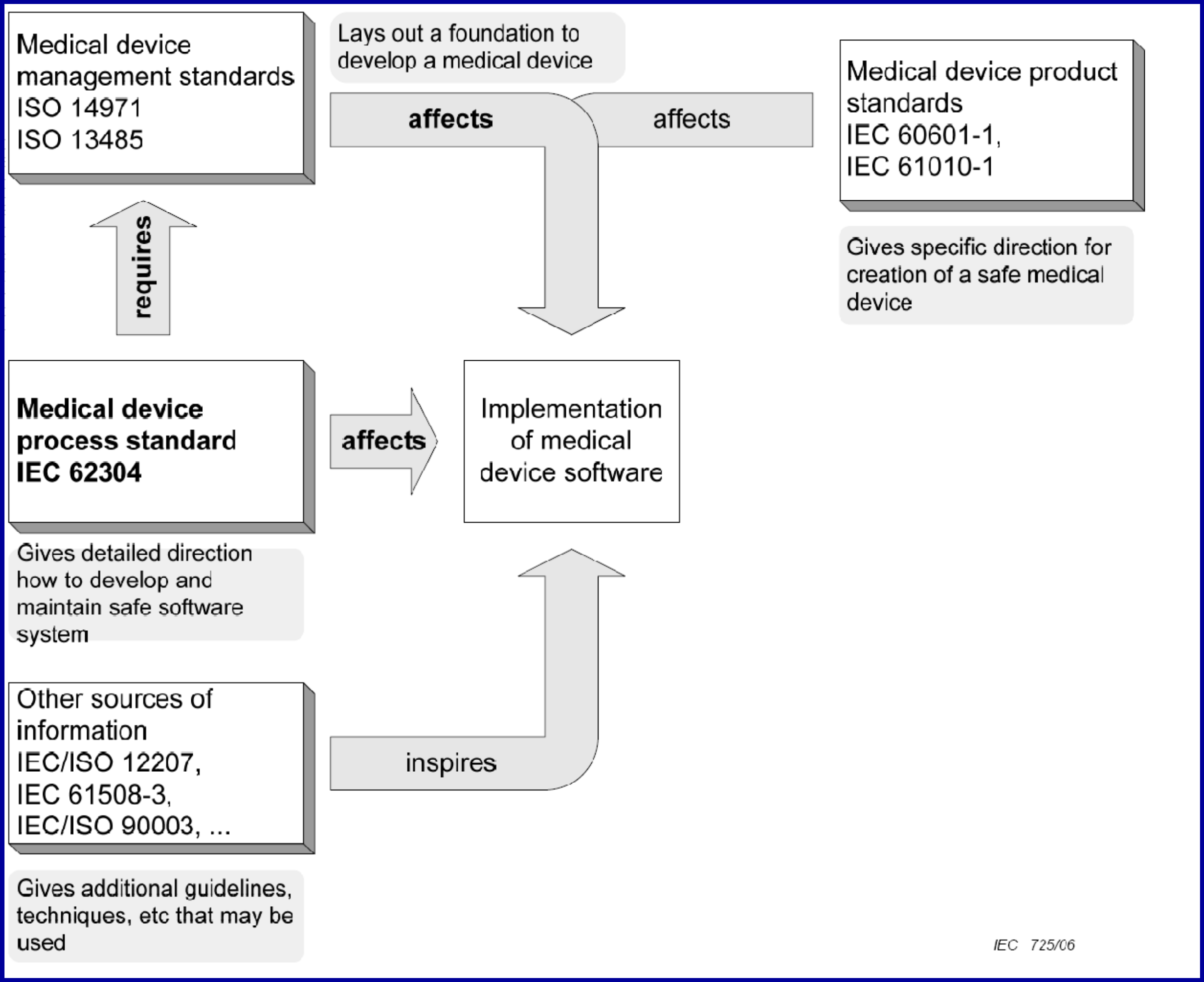
(Introduction, Figures 1 & 2)

Regulatory Context

Future of 62304

- Harmonization by EU
- Recognition by FDA

Relationship to Other Standards



Medical device management standards
ISO 14971
ISO 13485

Lays out a foundation to develop a medical device

Medical device product standards
IEC 60601-1,
IEC 61010-1

Gives specific direction for creation of a safe medical device

Medical device process standard
IEC 62304

Implementation of medical device software

Gives detailed direction how to develop and maintain safe software system

Other sources of information
IEC/ISO 12207,
IEC 61508-3,
IEC/ISO 90003, ...

Gives additional guidelines, techniques, etc that may be used

IEC 725/06

Traceability Tables
Annex C

- IEC 62304 vs. ISO 13485
- IEC 62304 vs. ISO 14971
- IEC 62304 vs. IEC 60601-1:2005
- IEC 62304 vs. IEC 60601-4:2005
- IEC 62304 vs. ISO 12207

Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices	IEC 62304
Level of Concern	Software Safety Classification (4.3)
Software Description	Software Requirements Analysis (5.2)
Device Hazard Analysis	Analysis of Software Contributing to Hazardous Situations (7.1)
Software Requirements Specifications	Software Requirements Analysis (5.2)
Architecture Design Chart	Software Architectural Design (5.3)

Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices	**IEC 62304**
Software Design Specifications	Software Detailed Design 5.4
Traceability Analysis	Throughout IEC 62304, including; 5.1.1, 5.2.6, 5.7.4, 7.3.3, 8.2.4
Software Development Environment Description	Software Development Plan 5.1
Verification and Validation Documentation	Throughout IEC 62304, including; 5.2.6, 5.3.6, 5.4.4, 5.5.5, 5.6.3, 5.6.7, 5.7.5, 7.3.1, 9.7, 9.8
Revision Level History	Configuration Staus Accounting 8.3
Unresolved Anomalies	Maintain Records of Software Problem Resolution 9.5

Questions

- What additional needs do you have?
  - Educational Materials
  - Tools
  - Policy Statements

Contact Information

- John Murray
  - Phone: (240) 276-0284
  - john.murray@fda.hhs.gov