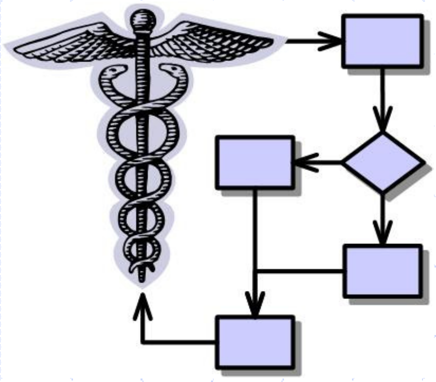
What is Validation?

Brian Shoemaker

ShoeBar Associates



Who I am

- Originally an analytical chemist
- 15 y in clinical diagnostics (immunoassay): analytical support → assay development → instrument software validation
- 6 y as SW quality manager (5 in clinical trial related SW)
- 8 y as independent validation consultant to FDA-regulated companies – mostly medical device
- Active in: software validation, Part 11 evaluation, software quality systems, auditing, training

Who are you?

- Where do you work – and what is your job role?
- What do you hope to get from this workshop
- Assumption: the software development expertise is with ***you*** in the audience

- Ground rules: you cannot just sit and listen!
- Every speaker has a different approach to this topic; I may miss your favorite area, since I’ve had to choose what to cover

Thesis

To serve its purpose, software validation cannot be an extra or an add-on, but must be built directly into the development cycle.

What is Validation?

- Validation myths cloud its usefulness
- *We verify software as part of design control*
- *Validation: know the user and his/her job!*
- *Always aim for safety*
- *Document – but choose your own lifecycle*
- *OTS / process software: we're still responsible*
- *Maintain the validated state*
- *Good engineering first – compliance follows*

Manager Beliefs – True or False?

- Validation is just to create documents
- Validation adds no value
- Validation is mysterious
- Validation comes after all development is done
- If development goes long, just cut the time for validation and everything will be fine

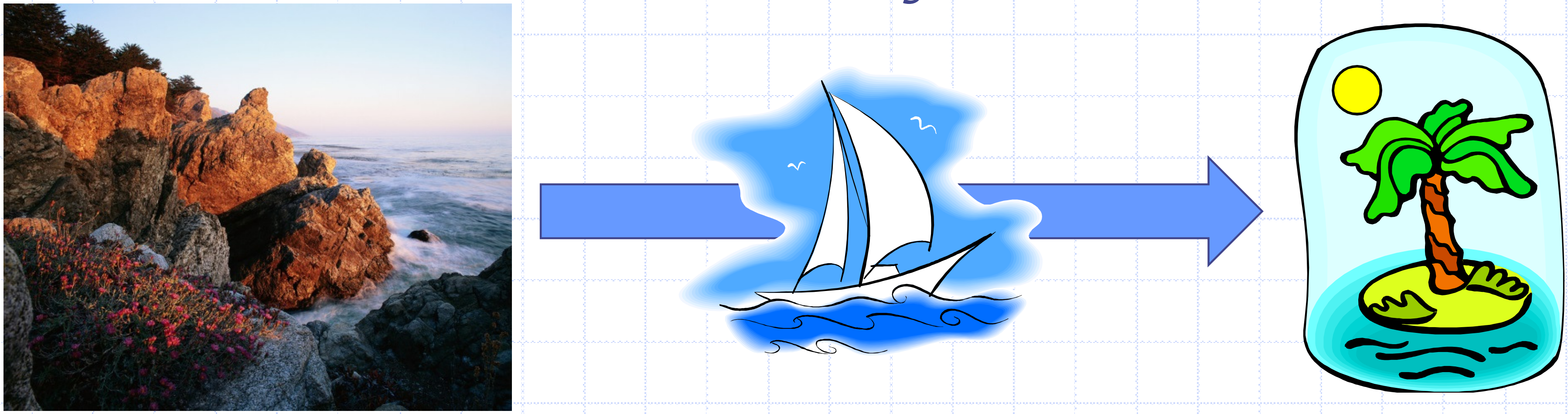
Developer Attitudes . . .

- My code runs perfectly
- Writing high quality code takes too long
- The UI doesn't matter
- I know perfectly what the end user wants / needs
- Finishing code for that key feature at 2 AM the night before the deadline is heroic

Just call it "V&V"

- What do we mean by "verification"?
- What do we mean by "validation"?
- Does the distinction matter?
- How do we know what to verify or validate?
- What about purchased code or production / quality tools? Do we validate those?

What’s the goal?



Unvalidated

Requirements
Design

Validated

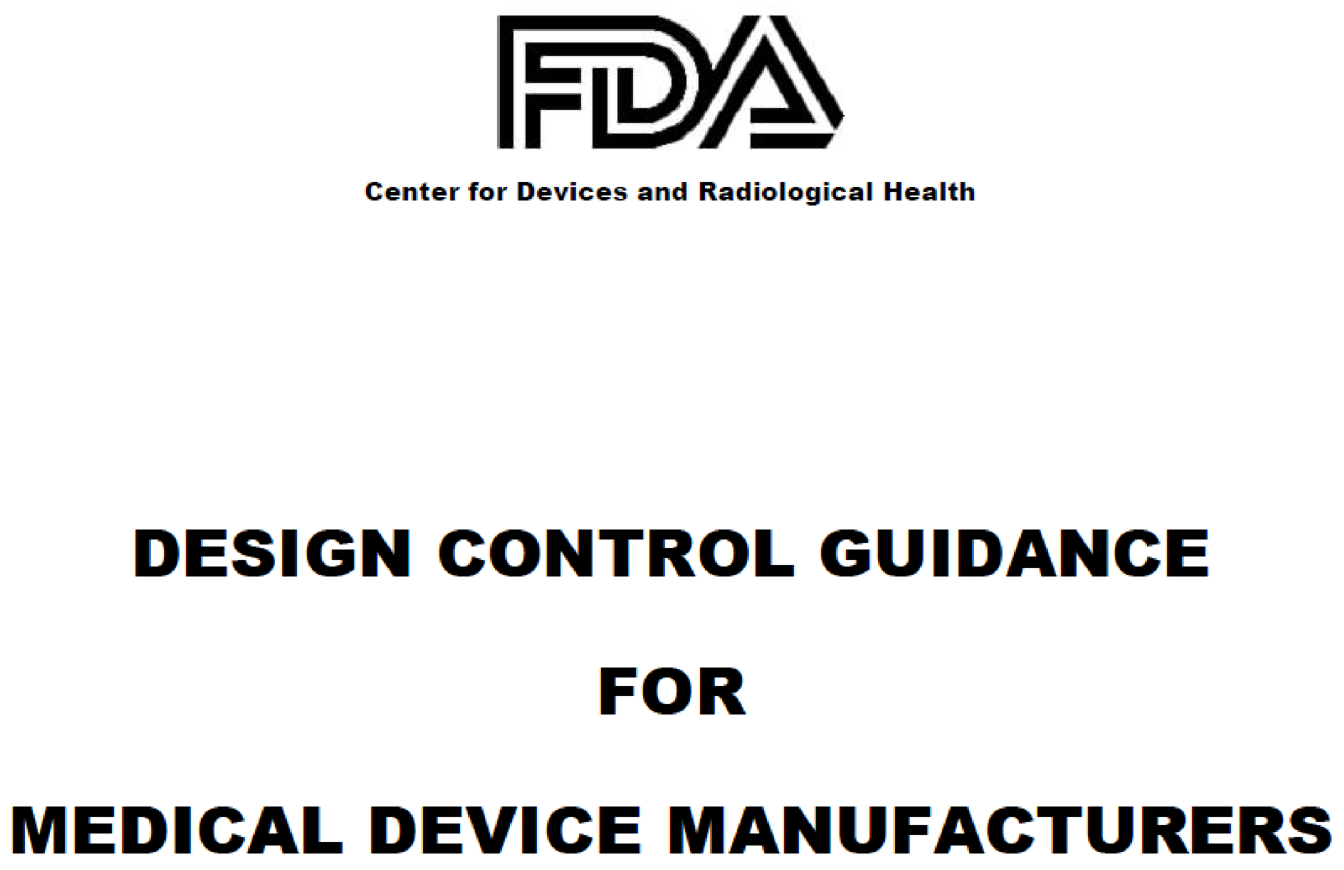
Testing
Traceability

We want to STAY here!

What is Validation?

- *Validation myths cloud its usefulness*
- We verify software as part of design control
- *Validation: know the user and his/her job!*
- *Always aim for safety*
- *Document – but choose your own lifecycle*
- *OTS / process software: we're still responsible*
- *Maintain the validated state*
- *Good engineering first – compliance follows*

Exercise: What is Design Control?



Center for Devices and Radiological Health

DESIGN CONTROL GUIDANCE

FOR

MEDICAL DEVICE MANUFACTURERS

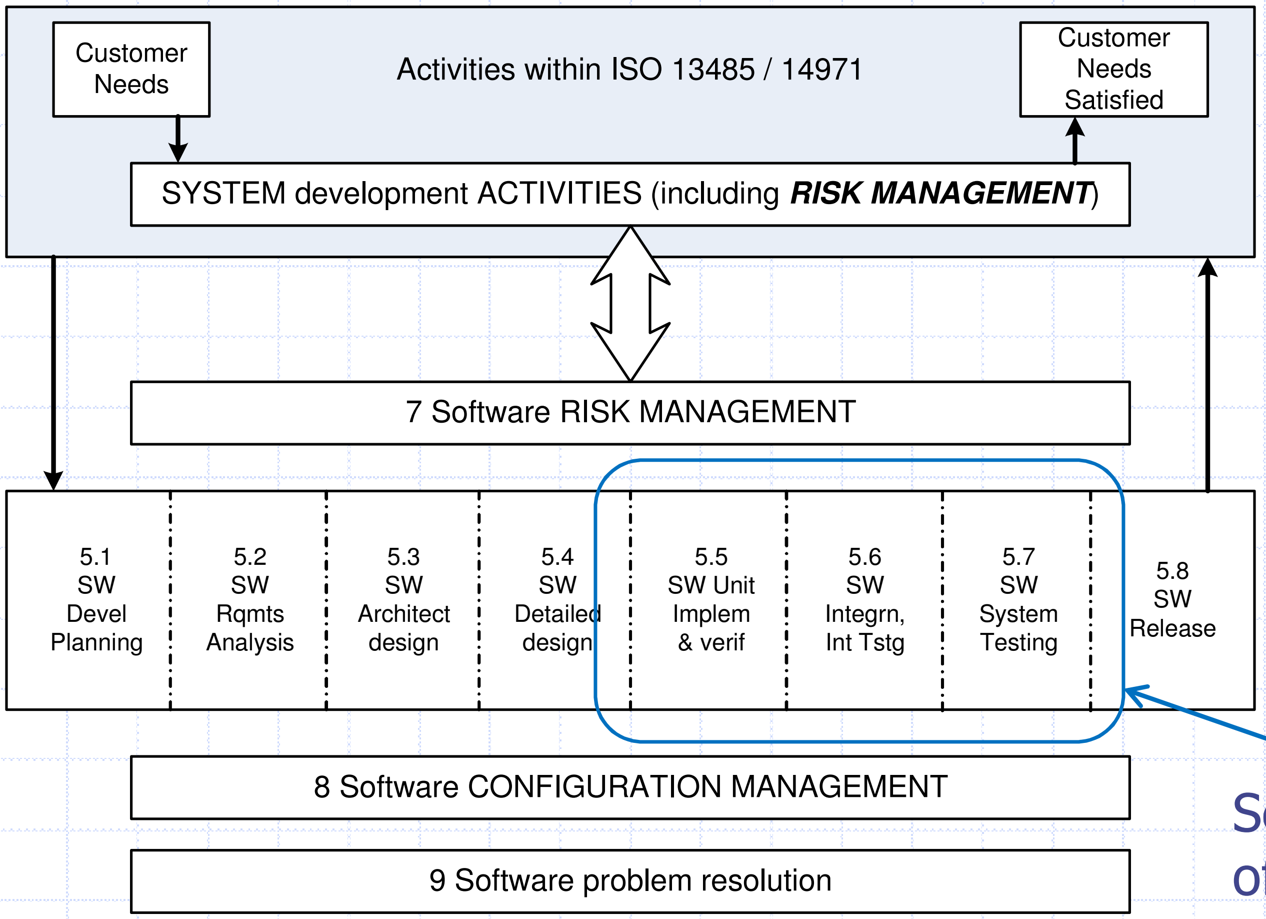
Design Controls, cont.

- Design controls are an interrelated set of practices and procedures that are incorporated into the design and development process, i.e., a system of checks and balances.
- Design controls make systematic assessment of the design an integral part of development. As a result, deficiencies in design input requirements, and discrepancies between the proposed designs and requirements, are made evident and corrected earlier in the development process.
- Design controls increase the likelihood that the design transferred to production will translate into a device that is appropriate for its intended use.

Reason for Design Controls

- How do we know what to develop?
  Where does this information ultimately come from?
- Who has ever developed a piece of software for which you knew everything you needed before you started?
- Who has ever developed a piece of software that ran perfectly as soon as it was finished, and pleased the customer / user?
- How do we ensure we've built the system we said we would build?
- Given the above, what then is the purpose of software **verification**?

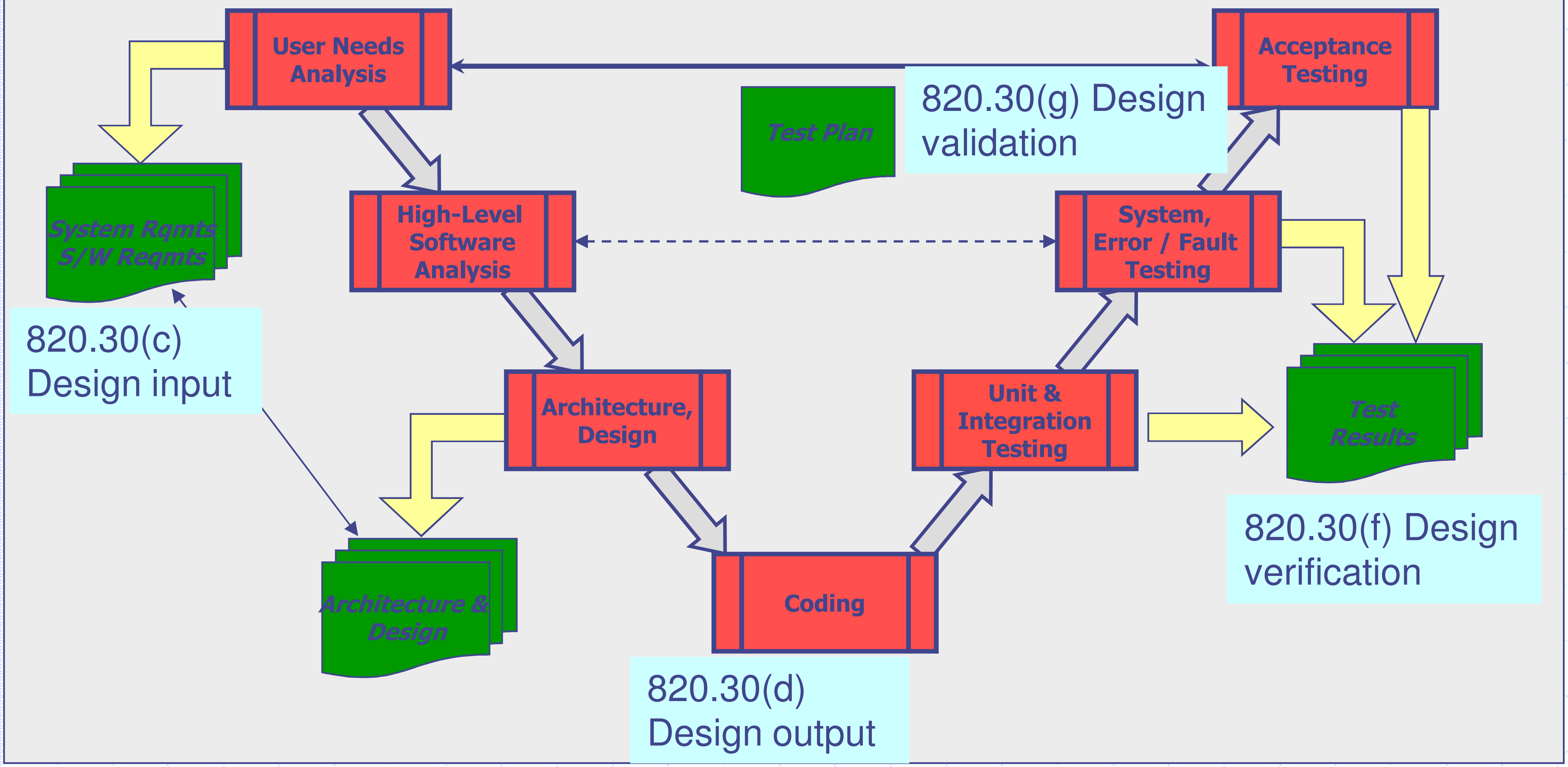
IEC62304: where is S/W verification?


Customer Needs
Activities within ISO 13485 / 14971
Customer Needs Satisfied
SYSTEM development ACTIVITIES (including ***RISK MANAGEMENT***)
7 Software RISK MANAGEMENT
5.1 SW Devel Planning
5.2 SW Rqmts Analysis
5.3 SW Architect design
5.4 SW Detailed design
5.5 SW Unit Implem & verif
5.6 SW Integrn, Int Tstg
5.7 SW System Testing
5.8 SW Release
8 Software CONFIGURATION MANAGEMENT
9 Software problem resolution

Several kinds of verification!

Put V&V in Context of Part 820

Design Control (820.30); 820.30(b) Design & development planning



820.30(c) Design input

820.30(d) Design output

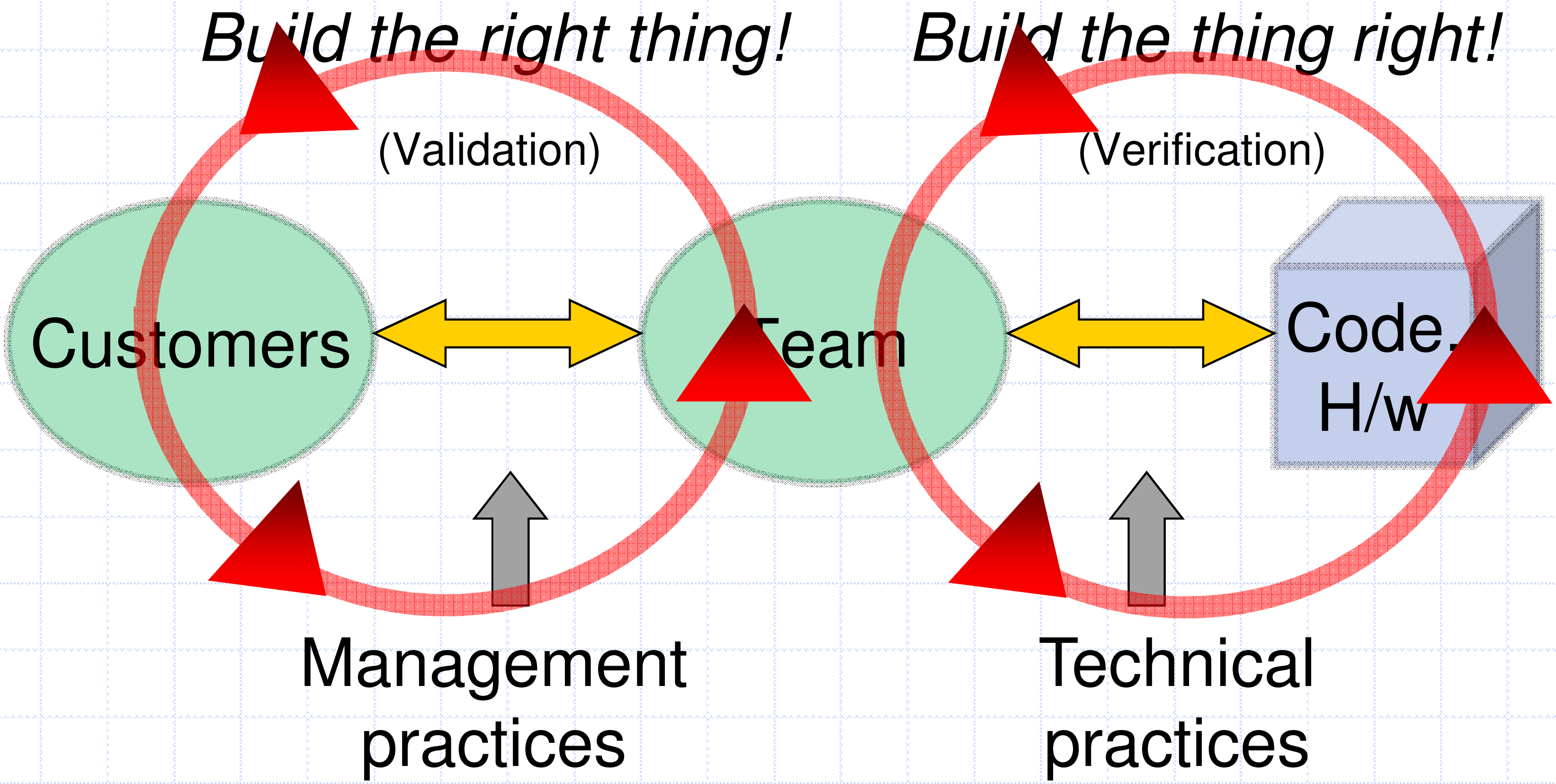
820.30(g) Design validation

820.30(f) Design verification

So where does verification begin?

- What do we need to capture?
- How do we find out?
- Is there an alternative way?

Consider an alternative



Source: N. Van Schooenderwoert

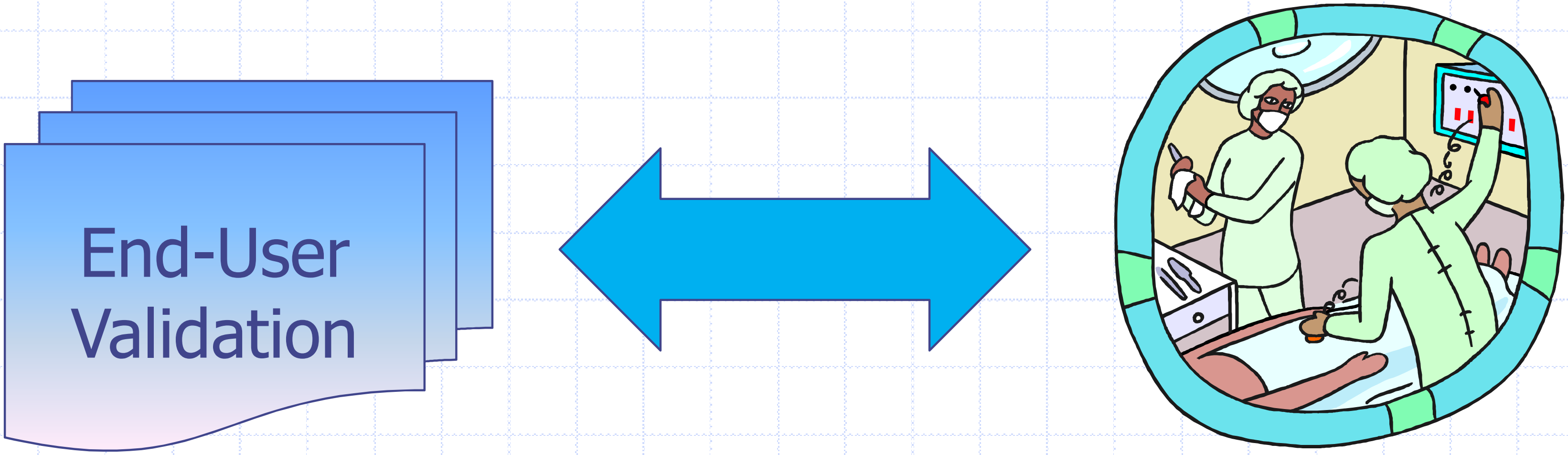
What are the documents?

- 21 CFR Part 820 Quality Systems Regulation
- 21 CFR Part 814 Pre-market approval of Medical Devices
- Design Control Guidance for Medical Device Manufacturers
- General Principles of Software Validation; Final Guidance
- Guidance for Off-The-Shelf Software Use in Medical Devices

What is Validation?

- *Validation myths cloud its usefulness*
- *We verify software as part of design control*
- Validation: know the user and his/her job!
- *Always aim for safety*
- *Document – but choose your own lifecycle*
- *OTS / process software: we're still responsible*
- *Maintain the validated state*
- *Good engineering first – compliance follows*

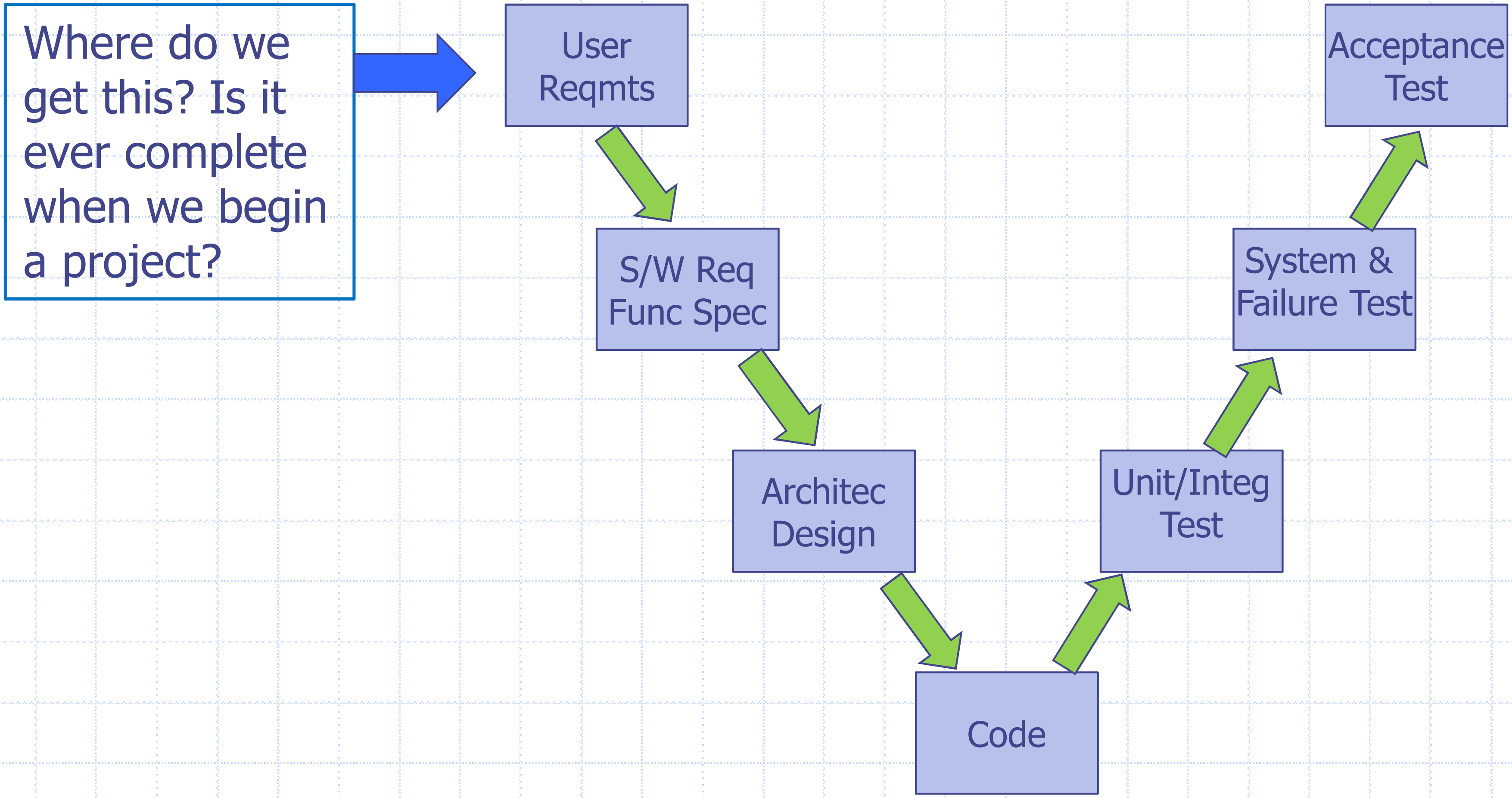
Two closely associated topics



End-User Validation

Human Factors Engineering

The V Model, again



How do we get the information?

- Ask questions – interview users

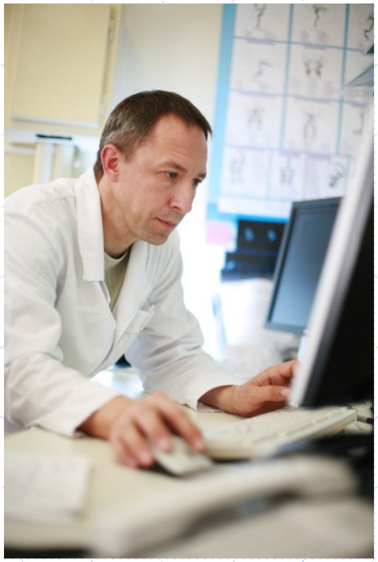
- Observe users

- Discuss / Brainstorm with users

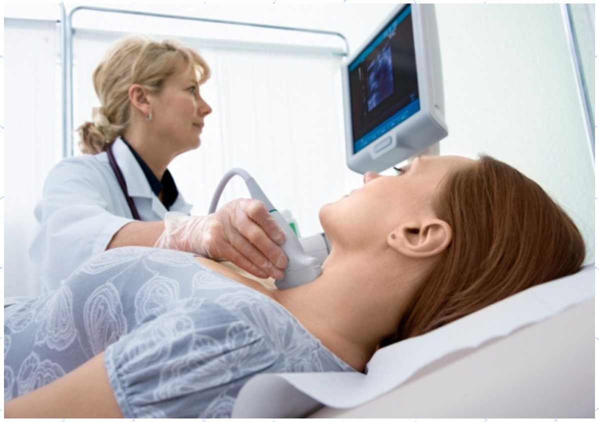
- Storyboard

- Review adverse event reports

Who are you designing for?



- Who will actually operate your system?
- Do you know what jobs they have to do every day? Where and under what conditions?



- What will make the device you're designing better than the one they're already using?
- How will you ever really know whether you've met their needs?
- **Could they misuse the system in a way that would hurt or kill the patient, the user, or a bystander?**

Documents - Usability

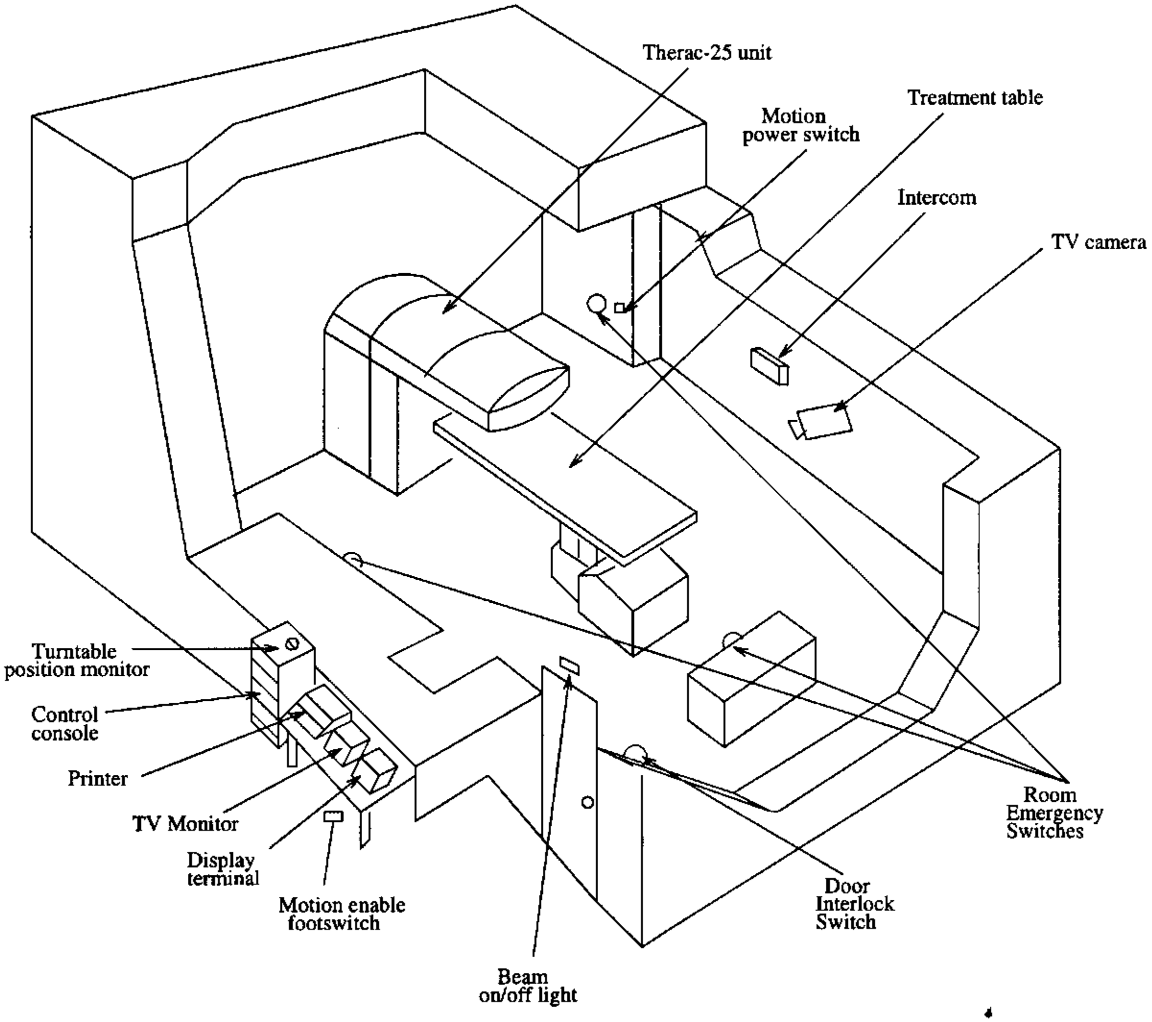
- Do It By Design - An Introduction to Human Factors in Medical Devices, Dick Sawyer, FDA, December 1996.
- Medical Device Use-Safety: Incorporating Human Factors Engineering into Risk Management, FDA, July 2000.
- Draft Guidance for Industry and Food and Drug Administration Staff - Applying Human Factors and Usability Engineering to Optimize Medical Device Design, FDA, June 2011.
- ANSI/AAMI HE75, 2009 Edition - Human factors engineering—Design of medical devices
- IEC 62366:2007, Medical devices -- Application of usability engineering to medical devices

What is Validation?

- *Validation myths cloud its usefulness*
- *We verify software as part of design control*
- *Validation: know the user and his/her job!*
- Always aim for safety
- *Document – but choose your own lifecycle*
- *OTS / process software: we're still responsible*
- *Maintain the validated state*
- *Good engineering first – compliance follows*

Can software hurt or kill?

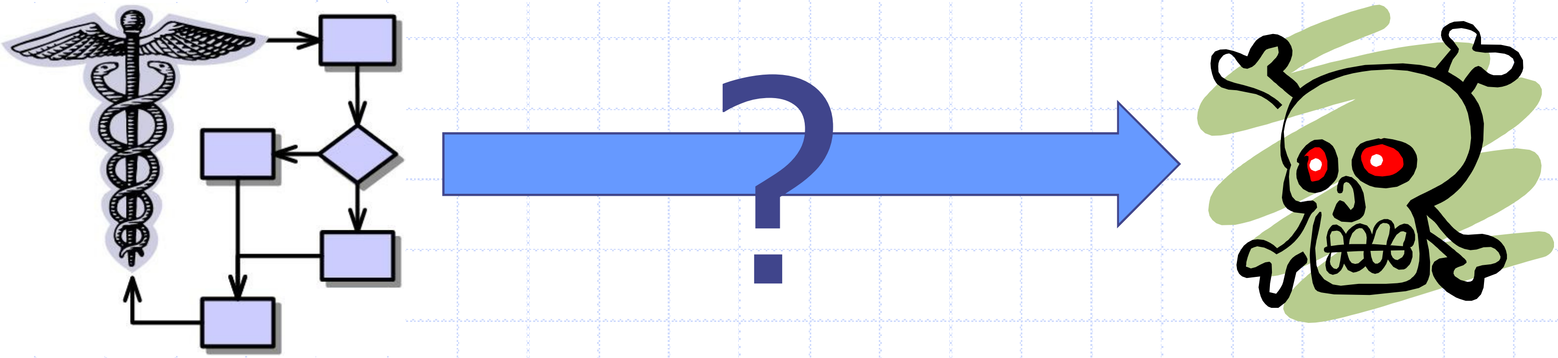
- Therac-25: cancer therapy system
- Six massive overdoses 1985-1987; 3 deaths
- Recalled in 1987



But what about today?

- May 08, 2013: (class 2) Electrosurgical unit; complaints of overheating, smoke vapors and failure to operate when used together with a specific model electrode.
- April 26, 2013: (class 2) Software correction for linear accelerator units to fix multiple safety-related issues (??)
- April 15, 2013: (class 1) Central programming unit for integrated IV medication safety system. Comm error on PC unit model 8015 with S/W ver 9.12 when $EtCO_2$ module or SpO_2 module is attached
- January 31, 2013: (class 1) Respirator w/ S/W vers ≤1.1.2; during ventilation of small pediatric patients with high airway resistance and low lung compliance, the ventilator O_2 consumption must be calculated using a larger margin than originally expected

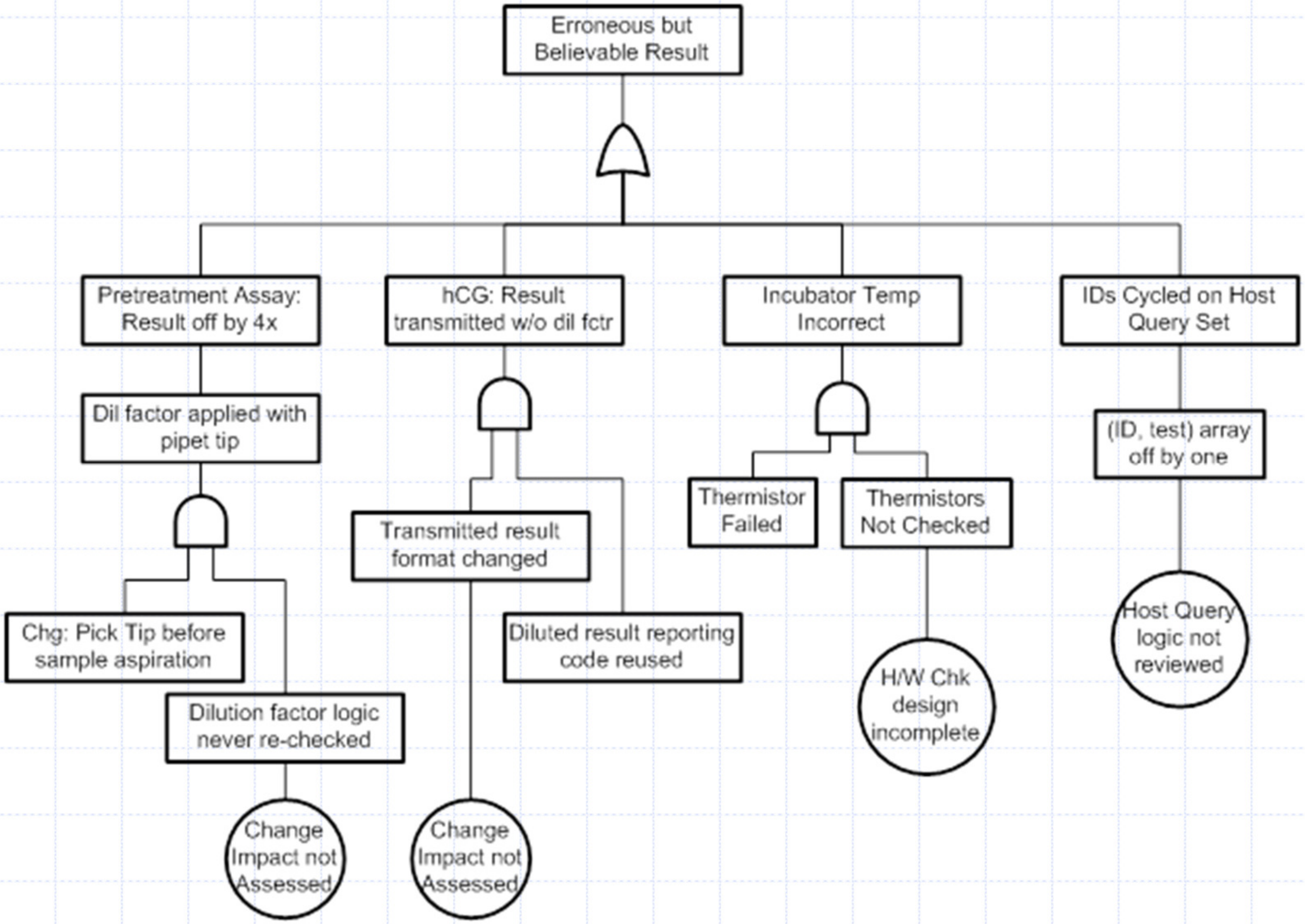
What is a software safety hazard?



Some ideas - sources?

• Direct failure	• *Incorrect algorithm / logic*
• Permitted misuse	• *No input checking*
• User Complacency	• *Inadequate warnings*
• User Interface confusion	• *Poor UI design, no validn*
• Security vulnerability	• *No attention to security*

Predict risks before design?


Erroneous but Believable Result
Pretreatment Assay: Result off by 4x
hCG: Result transmitted w/o dil fctr
Incubator Temp Incorrect
IDs Cycled on Host Query Set
Dil factor applied with pipet tip
Transmitted result format changed
Thermistor Failed
Thermistors Not Checked
(ID, test) array off by one
Chg: Pick Tip before sample aspiration
Diluted result reporting code reused
Host Query logic not reviewed
Dilution factor logic never re-checked
H/W Chk design incomplete
Change Impact not Assessed
Change Impact not Assessed

Assess risk within design?

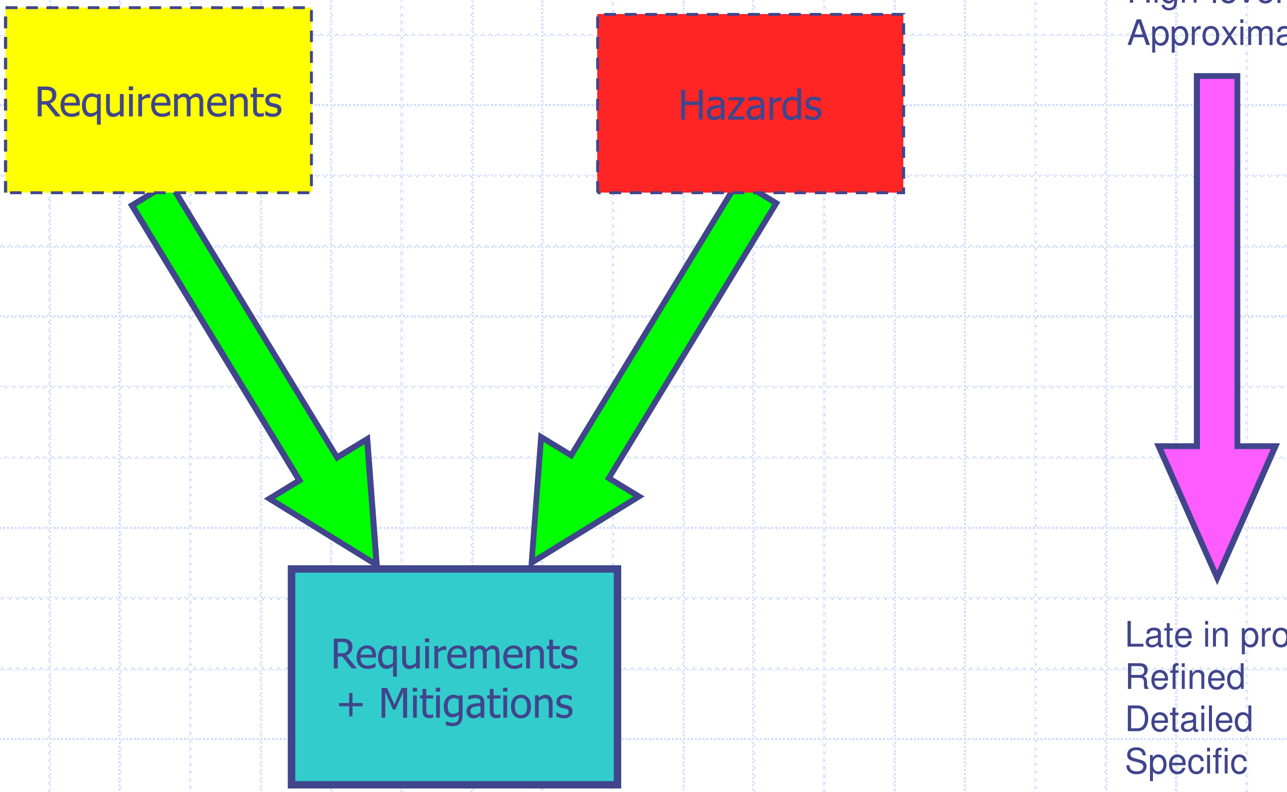
Failure Mode	Effect	Causes	S1	Mitigation	S2
Sample ID / results array off by one	Wrong results reported	Inconsistent array logic; incorrect initialization	5	Optional – operator approve results before saving	2
Initialization fails to warm up lamp	Can't perform analyses	Startup logic can be set to skip steps and left that way	4	Reset all startup parameters on initialization	1
Dilution factor associated with pipet tip, picked in advance	Wrong result reported (dilution applied to wrong sample)	Counting logic not rechecked when pick-in-advance process introduced	5	(a) Track dilution and pipet tip separately; (b) show dilution with reported result	1

S1 = Severity rating before mitigation; S2 = severity rating after mitigation

Severity (sample values only): 5 = critical; 1 = nuisance

Note this analysis does not include two of the standard engineering estimates: occurrence (probability) or detection.

Do we know all risks up front?



Requirements

Hazards

Requirements
+ Mitigations

Early in project
Preliminary
High-level
Approximate

Late in project
Refined
Detailed
Specific

Standards for Risk Management

- ISO 14971:2012 – Medical devices – Application of risk management to medical devices
- ANSI/AAMI/IEC 62304:2006 Medical devices – Software life cycle processes
- IEC 60601-1, 3rd edition Medical electrical equipment – Part 1: General requirements for basic safety and essential performance.
- AAMI TIR 32:2004 Medical Device Software Risk Management
- ANSI/AAMI/IEC TIR80002-1 Medical Device Software – Part 1: Guidance on the application of ISO 14971 to medical device software

What is Validation?

- *Validation myths cloud its usefulness*
- *We verify software as part of design control*
- *Validation: know the user and his/her job!*
- *Always aim for safety*
- Document – but choose your own lifecycle
- *OTS / process software: we're still responsible*
- *Maintain the validated state*
- *Good engineering first – compliance follows*

Thoughts on Validation Docs

- What does the FDA say about an activity that wasn't written down?
- What documents do we generate for device software development?
- Is there only one acceptable form for these?
- When must a requirements document be complete?

GPSV on Lifecycle

4.4 SOFTWARE LIFE CYCLE

Software validation takes place within the environment of an established software life cycle. The software life cycle contains software engineering tasks and documentation necessary to support the software validation effort. In addition, the software life cycle contains specific verification and validation tasks that are appropriate for the intended use of the software.

This guidance does not recommend any particular life cycle models – only that they should be selected and used for a software development project.

IEC 62304 on Lifecycle

Introduction

This standard does not prescribe a specific life cycle model. The users of this standard are responsible for selecting a life cycle model for the software project and for mapping the processes, activities, and tasks in this standard onto that model.

Annex B (informative)

Guidance on the provisions of this standard

The purpose of this standard is to provide a development process that will consistently produce high quality, safe medical device software. To accomplish this, the standard identifies the **minimum** activities and tasks that need to be accomplished to provide confidence that the software has been developed in a manner that is likely to produce highly reliable and safe software products. (...)

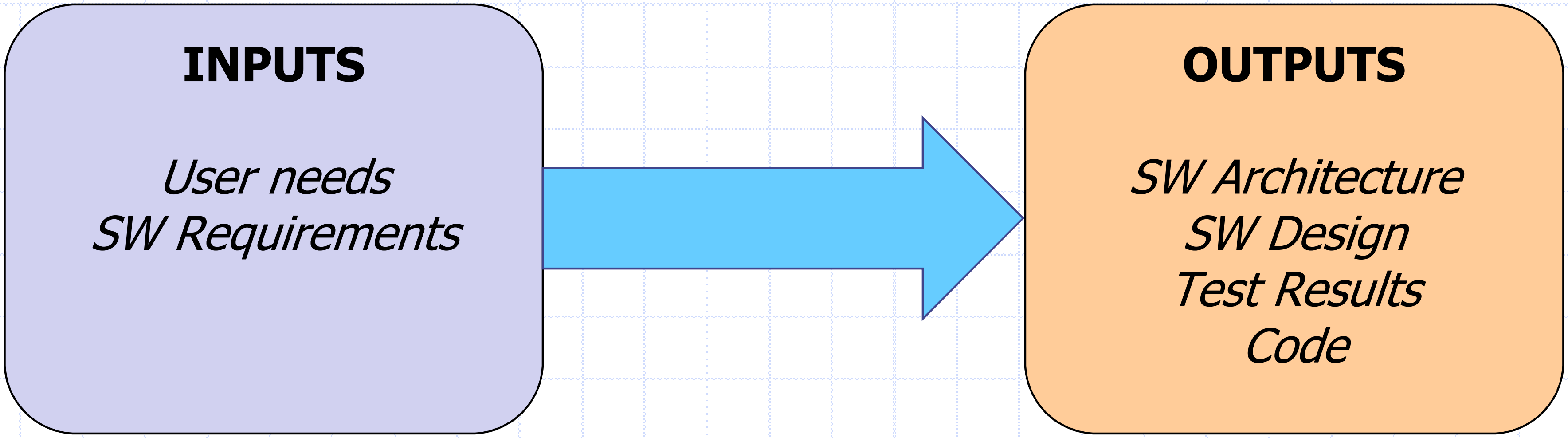
IEC 62304, cont.

Annex B (cont.)

This standard does not require a particular software development life cycle model. However, compliance with this standard does imply dependencies between processes, because inputs of a process are generated by another process. For example, the software safety classification of the software system should be completed after the risk analysis process has established what harm could arise from failure of the software system.

Because of such logical dependencies between processes, it is easiest to describe the processes in this standard in a sequence, implying a "waterfall" or "once-through" life cycle model. However, other life cycles can also be used.

Tell your story!



INPUTS

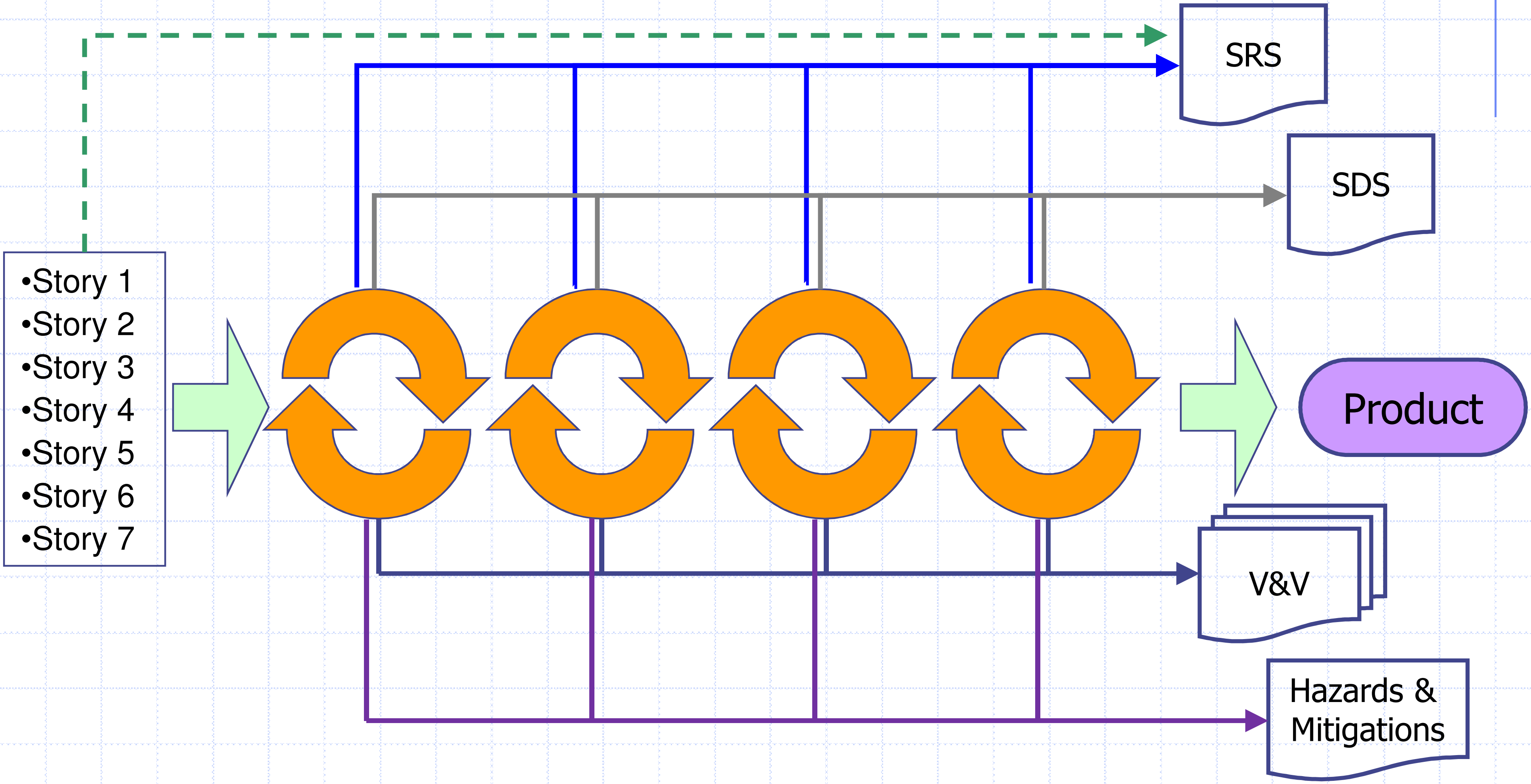
User needs
SW Requirements

OUTPUTS

SW Architecture
SW Design
Test Results
Code

Consider: Do good documents result if we try to write them after everything else is done?

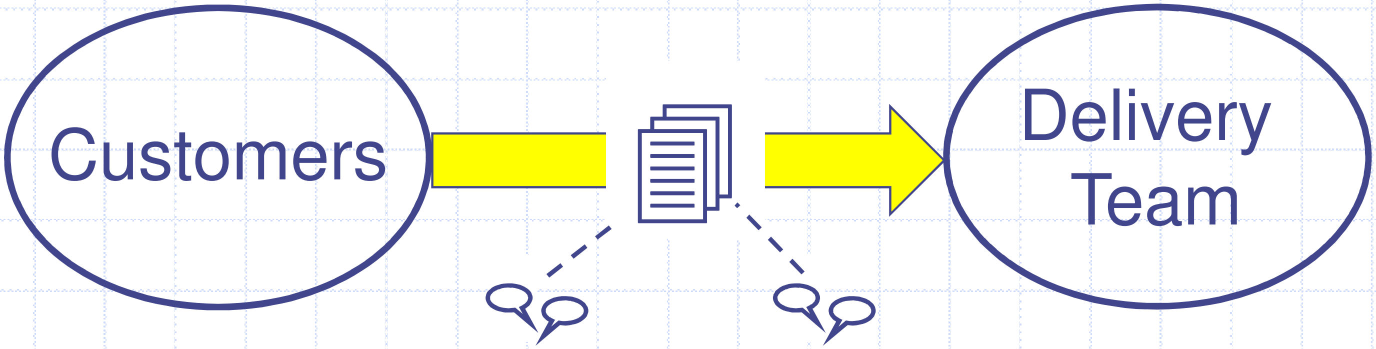
How best to document S/W?



ShoeBar Associates

How the best docs emerge

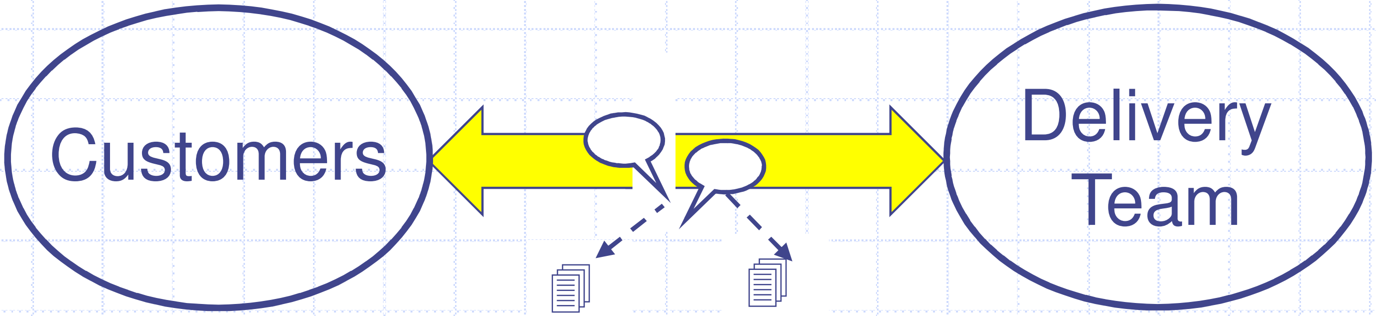
- Traditional: Documents, then conversations



Customers

Delivery Team

- Better: Conversations give rise to documents

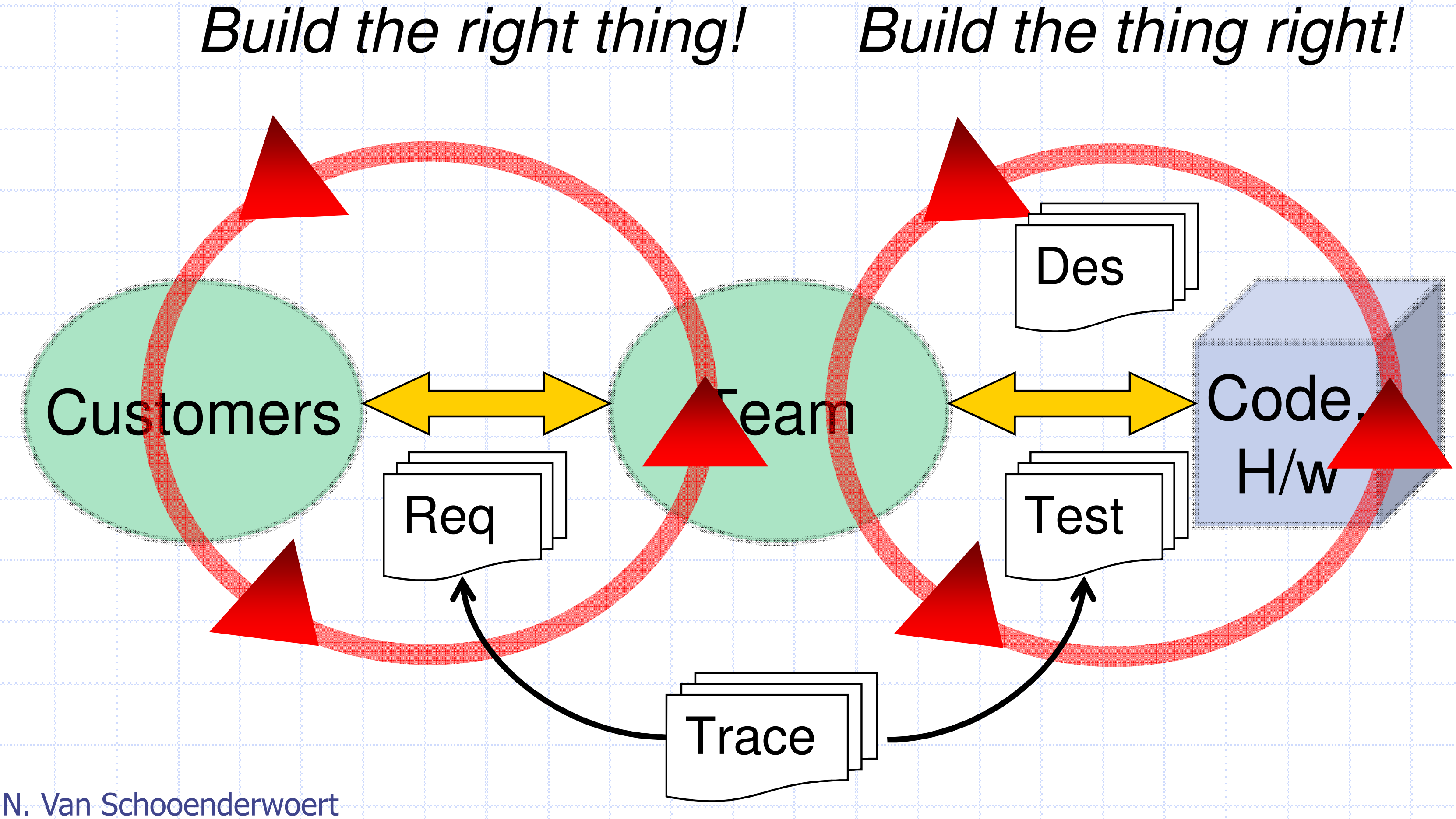


Customers

Delivery Team

Source: N. Van Schooenderwoert

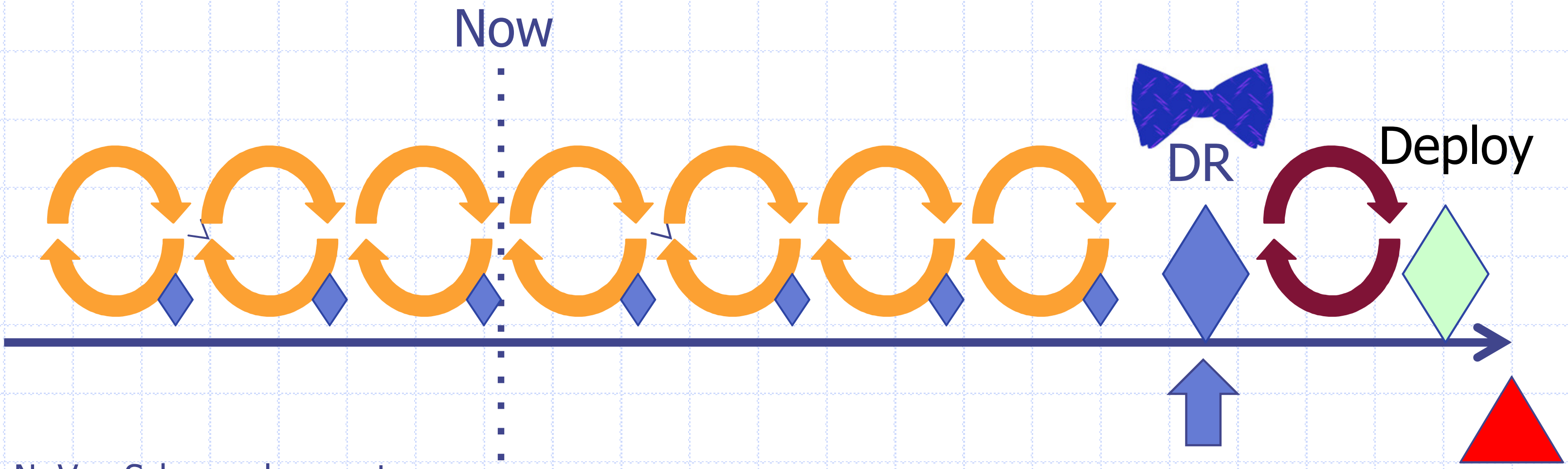
Documents - Collaboration



Source: N. Van Schooenderwoert

Don’t forget reviews

- Accept that we need to learn
- Each iteration has design, dev, test, demo (◆)
- We’ll hold the <u>formal</u> Design Review…



Source: N. Van Schooenderwoert

Guidances / Stds on Documentation

- 21 CFR Part 820 Quality Systems Regulation
- Design Control Guidance for Medical Device Manufacturers
- General Principles of Software Validation; Final Guidance
- Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices
- ISO 13485:2003 Medical devices - Quality management systems - Requirements for regulatory purposes
- ANSI/AAMI/IEC 62304:2006 Medical devices – Software life cycle processes

What is Validation?

- *Validation myths cloud its usefulness*
- *We verify software as part of design control*
- *Validation: know the user and his/her job!*
- *Always aim for safety*
- *Document – but choose your own lifecycle*
- OTS / process software: we're still responsible
- *Maintain the validated state*
- *Good engineering first – compliance follows*

Do you build *everything*?

Exercise: What external software is incorporated into your device?

- Device driver (e.g. I/O boards)
- Initialization parser
- Graphics library
- Math function library
- Logging utility
- Image processing package
- Barcode label reading or printing
- Client / server communications
- HL7 interface

Design / mfg / quality S/W?

- Enterprise resource planning
- Document management
- CAPA
- Metrology (gauge tracking / calibration)
- Configuration management
- Defect tracking
- Integrated development environment

What guides OTS validation?

- Key question to ask?

RISK!!

- OTS software: Guidance for Off-The-Shelf Software Use in Medical Devices
- Process / quality software: AAMI TIR 36:2007 Validation of software for regulated processes

What is Validation?

- *Validation myths cloud its usefulness*
- *We verify software as part of design control*
- *Validation: know the user and his/her job!*
- *Always aim for safety*
- *Document – but choose your own lifecycle*
- *OTS / process software: we're still responsible*
- Maintain the validated state
- *Good engineering first – compliance follows*

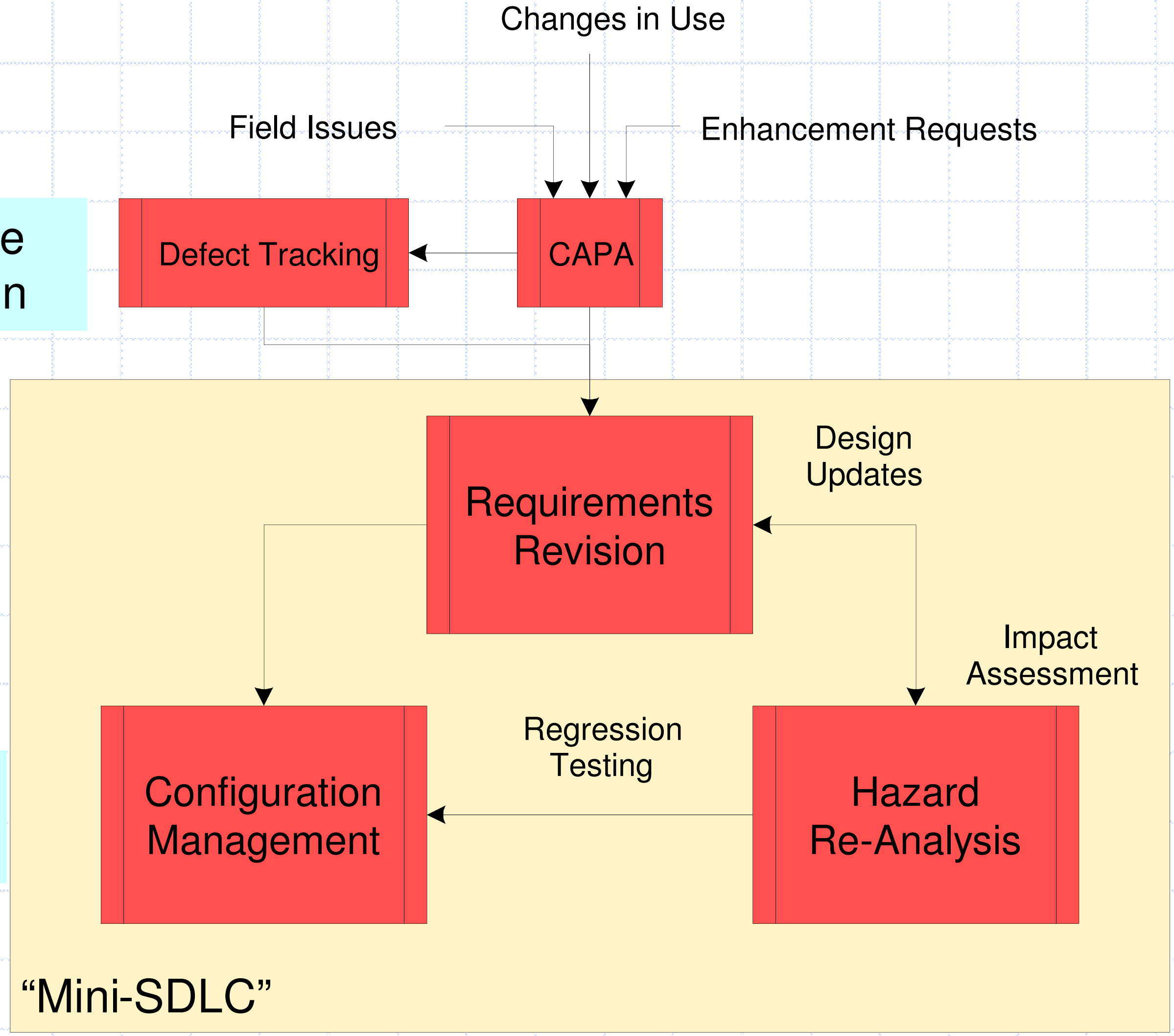
What happens after release?

- Do users ever find bugs in the field?
- Do you ever add enhancements?
- Do you release improved versions?
- How long can a medical device product be in service?

What we build, we must maintain

820.100 Corrective & preventive action

820.30(i) Design changes



What processes do we need?

- Change control / configuration management
- Defect tracking
- Document management
- Release management

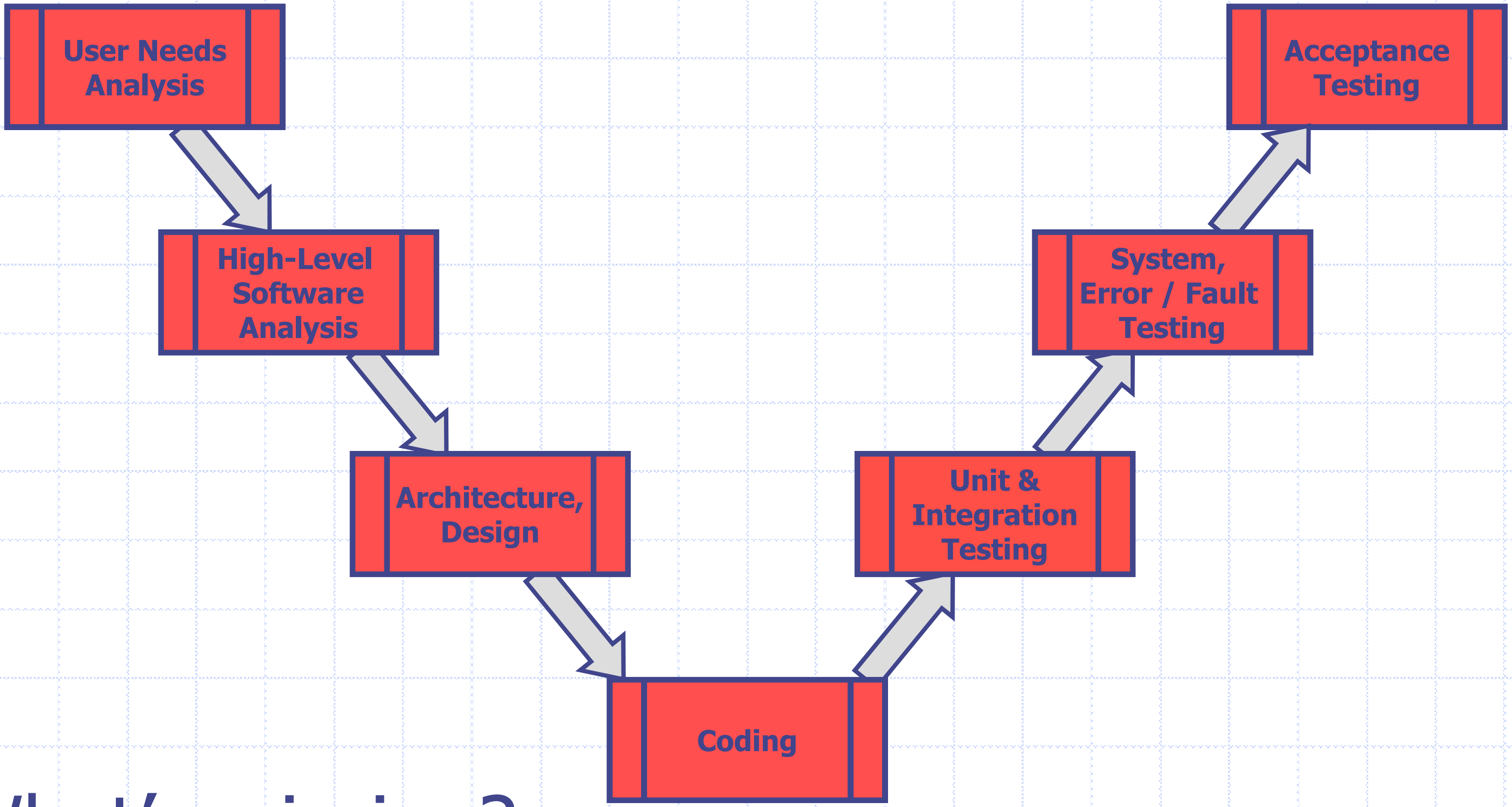
What is Validation?

- *Validation myths cloud its usefulness*
- *We verify software as part of design control*
- *Validation: know the user and his/her job!*
- *Always aim for safety*
- *Document – but choose your own lifecycle*
- *OTS / process software: we're still responsible*
- *Maintain the validated state*
- Good engineering first – compliance follows

Wrapup: What is Validation?

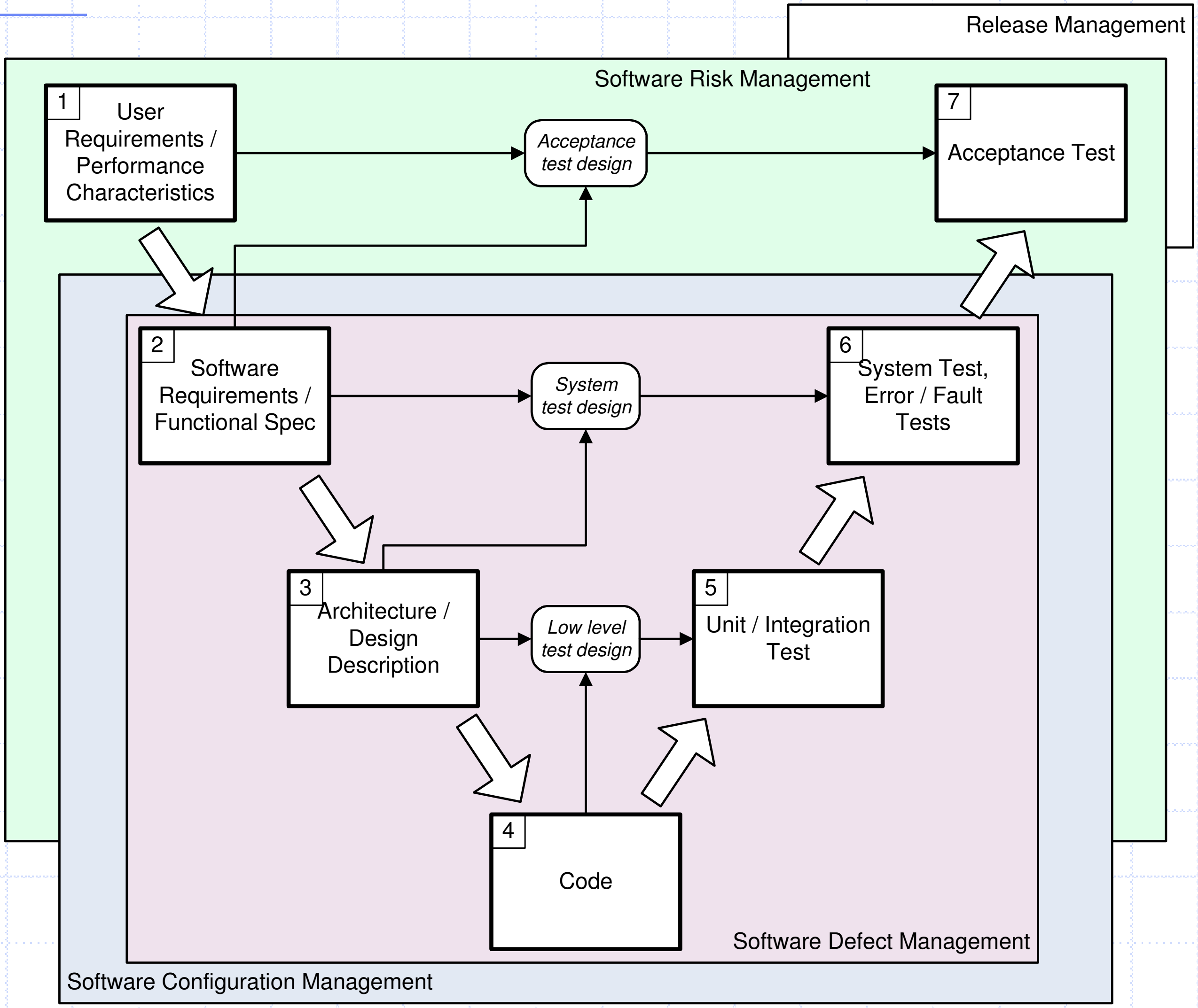
- Yes – I use the term two ways
- Where does validation begin? End?
- As we understand what a user needs, who else eventually needs that information?
- What does an ME generate to design a part, or an EE generate to design a circuit or cable?
- Are software practices fundamentally different from mechanical or electrical engineering?

What do we need?


User Needs Analysis
High-Level Software Analysis
Architecture, Design
Coding
Unit & Integration Testing
System, Error / Fault Testing
Acceptance Testing

- What’s missing?
- How are these boxes related to each other?

The Larger Picture


Release Management
Software Risk Management
1
User Requirements / Performance Characteristics
Acceptance test design
7
Acceptance Test
2
Software Requirements / Functional Spec
System test design
6
System Test, Error / Fault Tests
3
Architecture / Design Description
Low level test design
5
Unit / Integration Test
4
Code
Software Defect Management
Software Configuration Management

Parting thoughts

- Is validation the same as testing?
- Is software validation separate from development?
- Do we validate purely because it's required?
- Are we done when the product is released?

Contact information

Brian Shoemaker, Ph.D.
Principal Consultant, ShoeBar Associates
199 Needham St, Dedham MA   02026
781-929-5927
bshoemaker@shoebarassoc.com
http://www.shoebarassoc.com