Software Validation

Good Automation

Irving, Texas

GOOD AUTOMATION

WWW.GOODAUTOMATION.COM
(972) 717-4663
Sales@GoodAutomation.com

About Good Automation

- We develop custom automated test equipment (ATE) for medical devices
- We consult on Quality Management System (QMS) and FDA compliance
- Our deliverables include:
  - ATE software development
  - ATE software validation plans/protocols
  - Test method validation
  - Process validation
- We have strong capabilities in National Instruments (NI) platforms:
  - NI Certified LabVIEW Architects (CLA)
- Successful customer deployments:

GE Healthcare · Johnson & Johnson · Medtronic · Cochlear · Abbott

WWW.GOODAUTOMATION.COM
(972) 717-4663
Sales@GoodAutomation.com

Software VALIDATION- Why?

- Bad Reasons
  - Quality won't sign off unless I do it
  - The SOP and work instruction says I have to
- OK Reason
  - FDA's 21CFR820.70(i)-Automated Processes says I have to

Software VALIDATION- Why?

Medical Devices shall be Safe and Effective for their Intended Uses

- Prove that the manufacturing process does its part in mitigating risk to patient
- Prove that the test equipment involved does its part in mitigating risk to patient
- Prove that the ATE software does its part in mitigating risk to patient

SOFTWARE Validation- Why?

- 8-9% of 2015 FDA inspection observations (483s) are attributable to:
  - 820.75(a)- Process
  - 820.72(a)- Measurement
  - 820.70(i)- Software
- Both Processes and Measurement can be (and often have to be) automated
- When computers are used for Production or Quality System automation, 21CFR820.70(i) comes back into play
- So… why does the FDA care so much about software specifically?

SOFTWARE Validation- Why?

Hardware	Software
Quality is Fnc(R&D, Mfg)	Quality is only Fnc(R&D)
Less complex	More complex
Wears with use and time	Does not wear, time reveals bugs, ages like wine
Failures can be anticipated	No advanced warning for failures, latent defects, hidden in code paths can surface long after market release

- **"Because of its complexity, the development process for software should be even more tightly controlled than for hardware, in order to prevent problems that cannot be easily detected later in the development process." –FDA**
- Small isolated changes to code can cause problems elsewhere
- Given high demand for software professionals and highly mobile workforce, the developer who maintains software may not have been involved in the original software development. **Therefore, accurate and thorough documentation is essential**

-GPSV Section 3.3

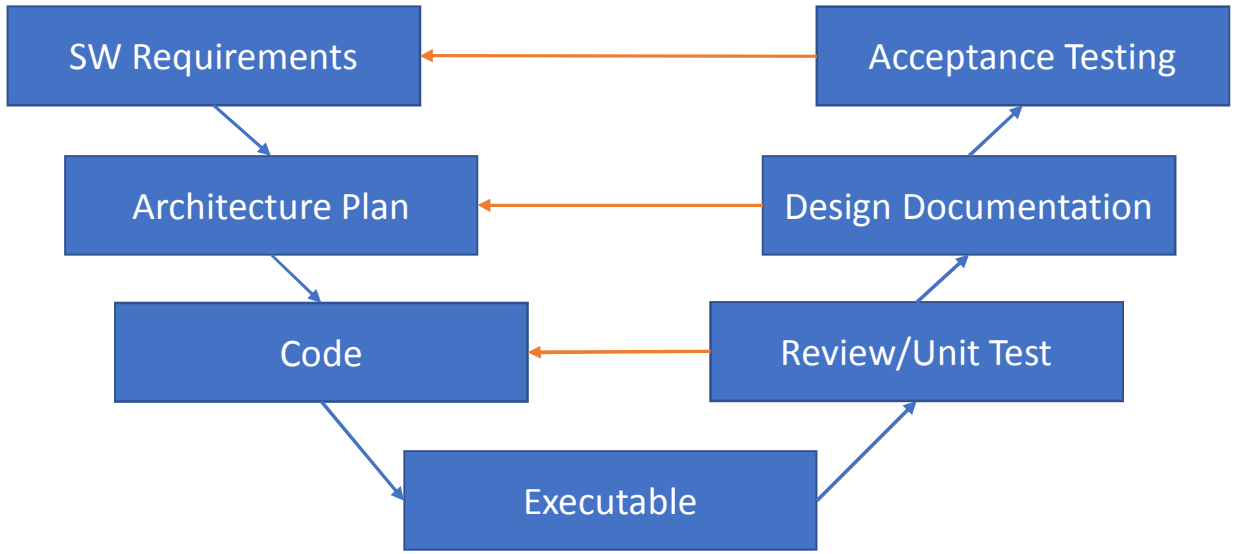
Software Validation Approach- Top-Down

- Top-Down approach is a Design technique
  - Start with the big ideas
  - Pick a part that is “too big” to be trivial
  - Decompose it into pieces
  - Repeat until everything is straightforward
- We should perform validation activities with the same intentionality- plan them along with the design efforts

Software Validation Approach- Top-Down

- Because validation is, itself, a designed (planned) activity: Top-Down Approach works for establishing validation activities
- Remedy to “Shoot-Ready-Aim”
  - Engineers tend to like diving straight into the low-level details
  - Throw away 10 drafts of the plan instead of one machine
- Planning interchange between design and validation activities ahead of time reduces schedule slip

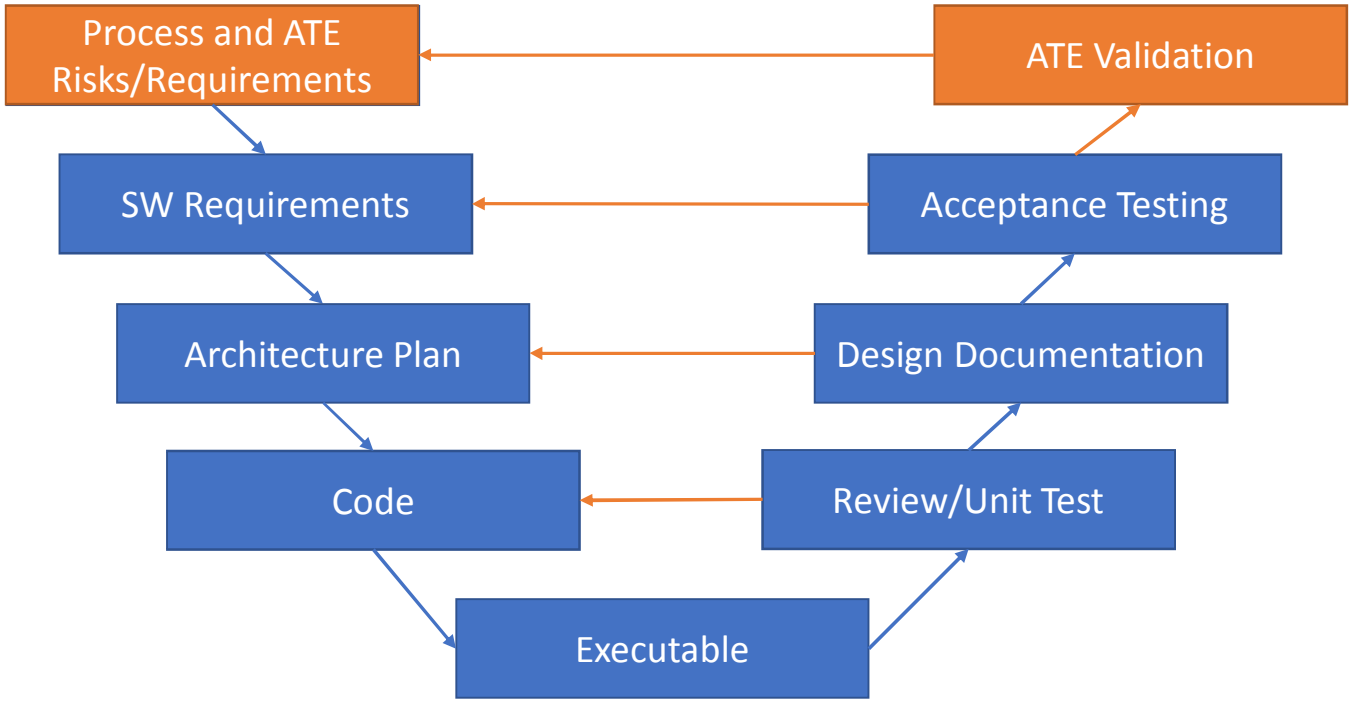
Software Validation Approach- Top-Down



Software Validation Approach- Risk-Based

- Medical devices shall be safe and effective for their intended use
  - Processes, ATE, software risks all point back to safety and effectiveness to the patient
- Assess **risk**
- Verify **mitigations**
- Collect objective **evidence**
- The end result of validation: objective **evidence** that unacceptable **risks** were **mitigated!**

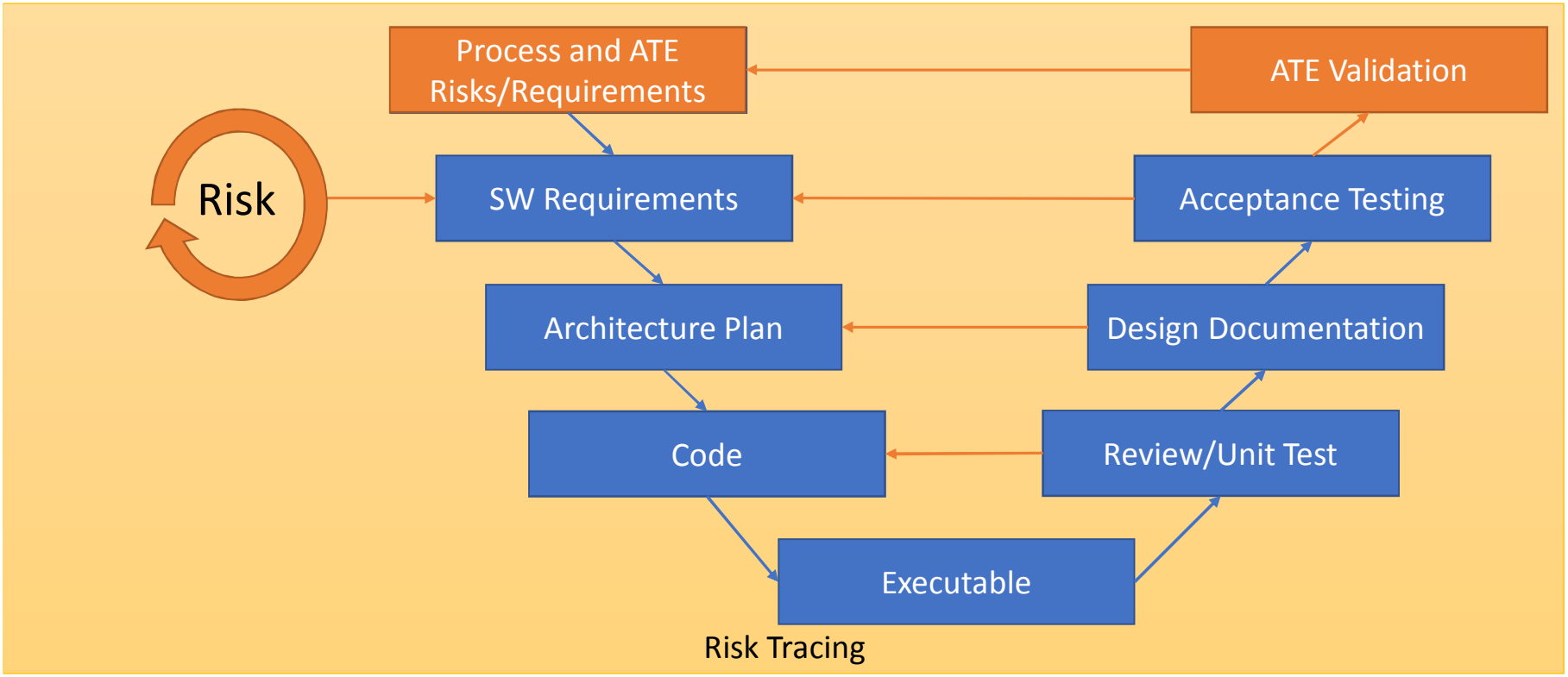
Software Validation Approach- Risk-Based



Risk is Iterative: Tracing

- (d/p/e/s)FME(C)A
  - Identify failure modes, define risk, establish mitigation strategy
- Trace early, trace often
  - Tracing done at **the appropriate step** will save much effort down the road
  - It can be as easy as this: “Covers: Risk/Requirement XXXX”…
- Trace provides evidence that all risk mitigations are fully implemented in code and fully verified by protocol

Software Validation Approach- Risk Iteration



Risk Tracing- A Brief Aside

- What about risk to machine operators? What about business risk? What about cybersecurity risk? What about…
- These should be handled by risk-based processes
- They **may** get covered by the same processes or by different processes
  - A team of engineers who understand patient safety may be different than a team of safety inspectors who handle ergonomics cases
  - It may all be the one person
  - Process should scale to your business/product

Software Validation Approach- Requirements

- Clear and concise conclusions to phrases “ATE shall…” or “Software shall…”
- Intentionally limited variety of verbs will aid in
  - Software modularity decisions
  - Acceptance Testing grouping
- Limit the use of “should” as this causes the programmer to decide what “should” be. Be explicit
- Cover: intended uses, risks, higher level requirements

Back to "Top-Down" Again…

- Everything up to this point done **before** design
- Some tasks cannot be done as well after development
  - For example: design requirements
  - Exception: Off the shelf, but that's not what runs your ATE
- Some tasks can be done, but at far greater time and effort (risk tracing)

Validation at Good Automation

- Risk Driven
- Multidisciplinary personnel
  - Validation content written by software developers
  - Internal reviewers understand needs for validation
  - Design for Validation
- Work within customers' quality standards
  - Experience rapidly ramping up on QMS requirements
  - Deliverables targeted at QMS requirements