Blackwood
Embedded
Solutions

Strategies for Medical Device Software Development

Presented By

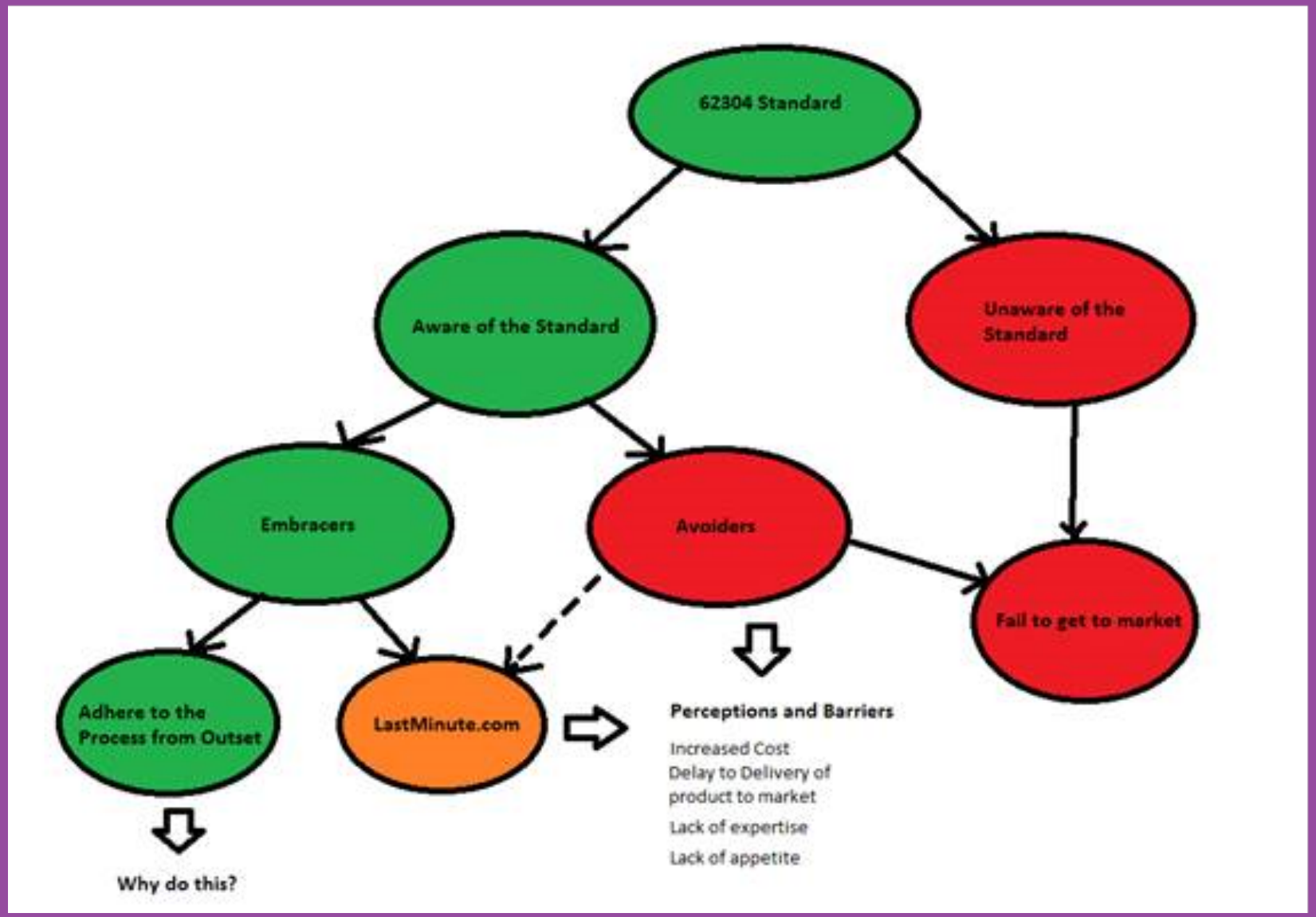
Anthony Giles of Blackwood Embedded Solutions Ltd

Introduction

Blackwood
Embedded
Solutions

- Standards – 60601-1 in particular clause 14, 62304 – Software development lifecycle, 14971 – Risk analysis
- Different Companies Views of standards – FDA guidance quoted “Least Burdensome approach”
- Key areas where mistakes are made
  - Software Safety Class
  - Life Cycle Model
  - Role of Prototypes
  - Modularising code
  - FMEA/Risk analysis
  - SOUP
  - Documentation
- Conclusion

Differing Company Views of standards



Software Safety Classification

Blackwood
Embedded
Solutions

- 62304 states

  "The software safety classes shall initially be assigned based on severity as follows:

  Class A: No injury or damage to health is possible [FDA –MINOR]

  Class B: Non-SERIOUS INJURY is possible [FDA – MODERATE]

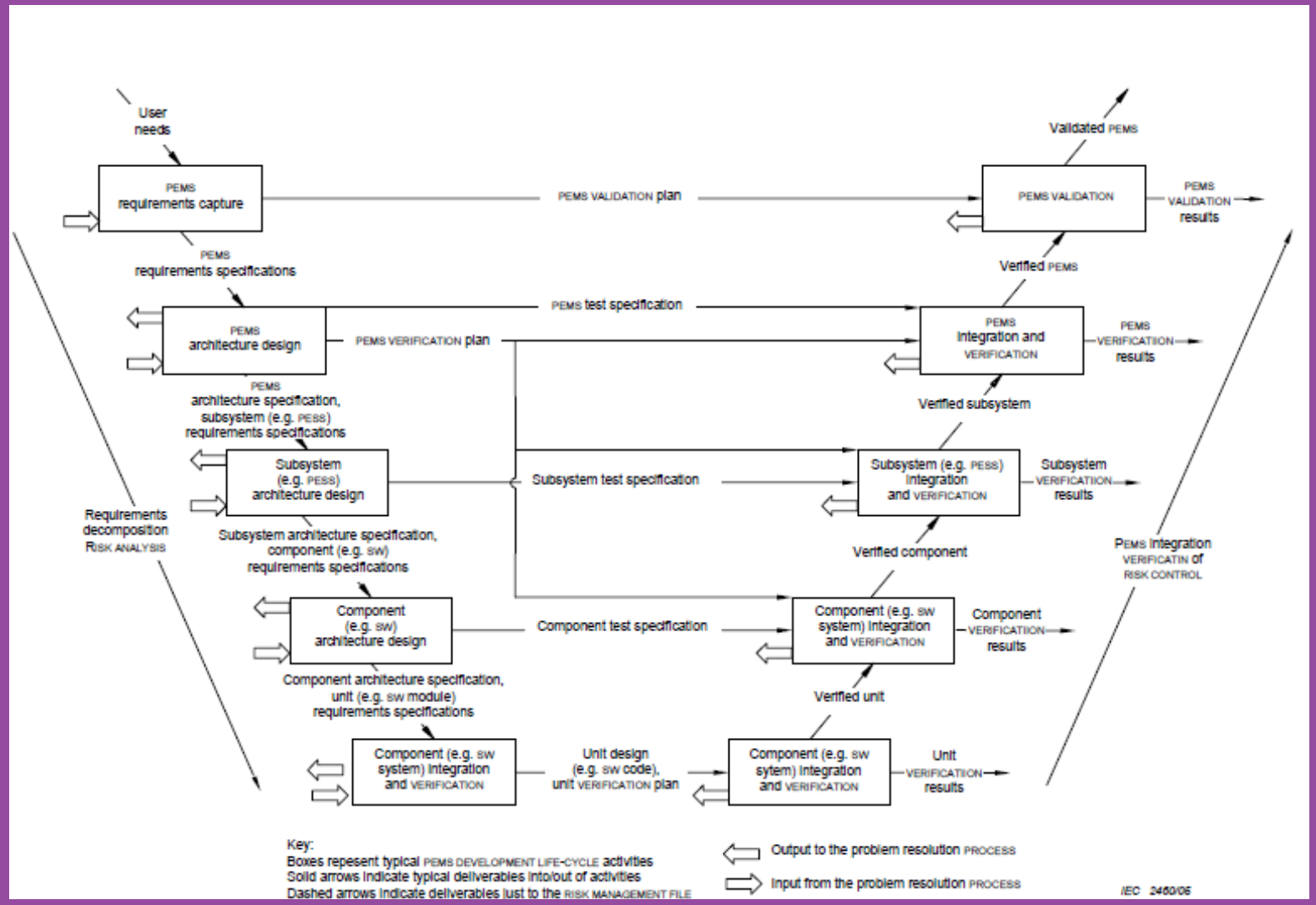
  Class C: Death or SERIOUS INJURY is possible [FDA – MAJOR]

  If the HAZARD could arise from a failure of the SOFTWARE SYSTEM to behave as specified, the probability of such failure shall be assumed to be 100 percent."

- **FDA "Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices"**
  http://www.fda.gov/MedicalDevices/DeviceRegulationandGuidance/GuidanceDocuments/ucm089543.htm
- Definition of what constitutes a serious injury
- Simple Yes/No Questionnaire to decide which category you fall in

- Common Mistake: - Avoid (We are class A) or leave until to late in the project

Software Lifecycle Model



- Doesn't have to be Waterfall
- Doesn't show how prototyping fits in
- More modern Agile approach is to specify part of the system that is known and run through the V process multiple times

Role of Prototypes

Blackwood
Embedded
Solutions

Some Pros

- Fast Proof of Concept
- Helps Discover Unknown Requirements or refine requirements earlier in the Project
- Help get an understanding of how best to code Modules/Units for particular Hardware

Cons

- Code almost never optimum architecture
- Almost never includes Risk Mitigation
- Almost never follows coding standards
- Leads to unstructured spaghetti code.
- Not well understood as a whole or well tested

- Common Mistake: Not knowing when to stop prototyping and start actual design
  - Lack of time/money to document test or perform Risk Mitigation
  - More buggy code
  - Unsafe code

Modularising Code

Blackwood
Embedded
Solutions

Breaking down complex designs in to smaller chunks

- Even the brightest people can hold no more than 6 individual thoughts in their heads at once
- Increases understanding of the workings of individual parts of the system
- Locks functionality into box so you only have to understand how to use it in the system
- Makes it easier to understand the complex system as interfaces between individual parts
- Makes it easier to document
- Makes it easier to test individual parts
- Makes it easier to apply Risk Analysis/FMEA
- More efficient code
- Less documentation
- Reduced Testing

- Common Mistake – Not breaking down code into modules leads to
  - More code to document
  - More testing
  - More buggy code
  - Unsafe code

Risk Analysis/FMEA

Blackwood
Embedded
Solutions

Benefits

- Makes code more robust
- Makes software safe
- Meets requirements of 14971

Cons

- Adds overhead to Processor resources time/memory
- Adds a lot of additional requirements to the project

- Common Mistakes: - Leaving to end of project
  - Lack of paperwork
  - Prototype left to long and to hard understand
  - Running out of processing resources
  - Not having room for additional safety requirements
  - Missing essential risks

SOUP – Software of Unknown Provenance

Blackwood
Embedded
Solutions

Benefits

- Save Time/Money using “Off the Shelf Code”
- Fast Prototypes

Cons

- Usually not designed to 62304
- The original coders don’t know your application and have not implemented Risk Mitigation
- Original requirements often not available to test against

- Common Mistakes: - Using SOUP with out understanding it
  - In some circumstances its impossible to test
  - Needs assessing for Risk in your product (even software already deemed safe)
  - Even if full source code available, will probably need rewriting to fix bugs, include risk measures
  - Supplier Documentation package and Test results sometimes available but at a high cost

Conclusion

Blackwood
Embedded
Solutions

- Avoiding the standard often leads to more complex designs
  - Increased Paperwork
  - Increased Testing
  - Increased Bugs - Safety issues
  - excessive project time and cost
  - Failure to get Regulatory Approvals and to market

- Embracing the Standards
  - They identify a lots of risk for you and even suggest solutions
  - 62304 Proven techniques to create bug free and faster to market solutions
  - Reduced cost and timescales
  - Almost Guaranteed Regulatory Approval

As the FDA say:

Embracing the standards is the “Least Burdensome Approach”