Testing in the Model-Based Development World

Boston Scientific

Delivering what's next.™

A case study of what it *really* takes to use MBD

David Cook, CSTE & Dr. Feng Zhu, PhD., CSTE

Boston Scientific CRM, Saint Paul, MN

Overview

We will discuss what is needed to implement testing in a model-based development (MBD) environment

Laying the Foundation

Framework Development

Test Development

Formal Verification

Costs & Benefits

Final Insights

Project Background

Complex medical device

- High regulatory scrutiny on process and safety

Highly complex software system

- ~2000 system requirements
- ~130 different modeled features
- ~5000 individual test cases
- 200,000+ lines of code (embedded)

Laying the Foundation

Boston Scientific

Common Architecture

- Auto-generation of code header files
- Code generation not necessary

Firmware Verification Support

- Verification test bus
  - Significant events (includes watchpoints)
  - Messages
  - Status changes
  - Debug notification

Product Line Engineering

- Support for varying device feature sets
- Ability to selectively test based on the variation model

Laying the Foundation

Boston Scientific

Tooling and Infrastructure

- Test systems that are device-equivalent and a test execution framework
- Verification behavior analyzer
- Automated model vs. firmware comparison
- Test script IDE

Organizational Support

- Full support from mid- and upper-management
- Team of developers to put together the tools

Framework Development: Isolating Failure modes

Boston Scientific

Test Station Support

- Networked test station pool created
- Test capability configuration
- Dynamic testing software loading
- Sharing testing resources among projects

Behavioral Comparison Error Analysis

- Comprehensive real-time behavior comparison
- Reporting mismatches automatically
- Auto and manual analysis

Syncing to the Real World

- Addressing timing drifting
- Isolation of real behavioral issues
- Root cause identification of behavioral violation

Framework Development: Tracking Common Problems

Infrastructure Support for Regression Execution

- Test job management
- Test submission and execution monitoring

Issue Reporting

- Automated reporting
- Profiling regression issues by type, feature, or testing phase
- Reports become input to Analysis Activity

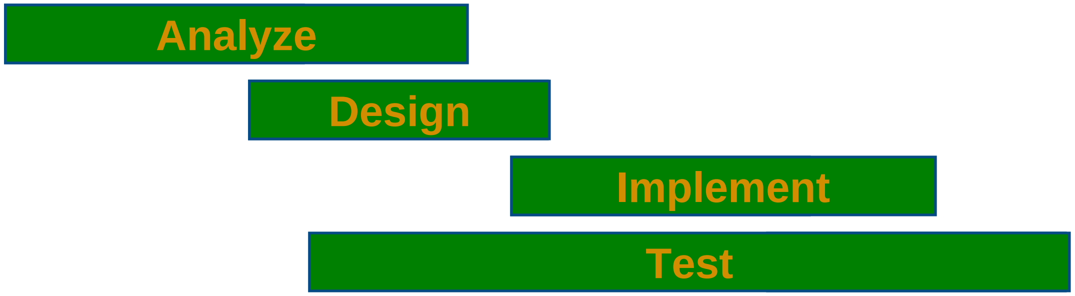
Issue Analyzing

- Common issue tracking
- Known issue mapping
- Automated analysis using issue patterns

Framework Development: Schedule and Time-Table

Typically, the project schedule may look like this:

With, MBD the schedule changes:



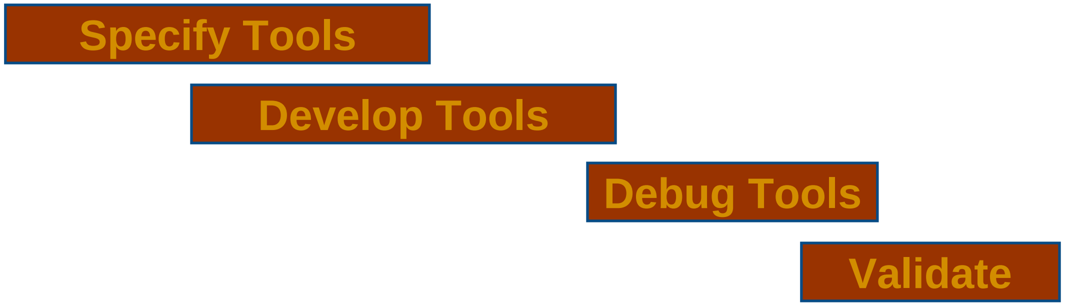
Analyze

Design

Implement

Test

And, in terms of the framework you have effort:



Specify Tools

Develop Tools

Debug Tools

Validate

This slide does intend to in any way promote or endorse the waterfall model, no waterfalls were harmed as a result of the development described herein

Test Development: Early Test Development

Boston Scientific

Gaining the Knowledge

- Early involvement in Requirement analysis
- Expertise in reading the model and prepare for testing

Using the Model

- Preliminary model testing
- Develop high level test cases
- Find requirement issue early

Pitfalls to Starting Early

- Requirement/Model continuously in flux
- Need good model turnaround time
- Plan for formal architecture update/build/release cycles
  - Many teams depend on the architecture for varying needs

Test Development: Test Station Integration

Boston Scientific

FW versus Model

- Timing difference between model and FW execution
- Test system injected issues
- Test update needed

Proving Scenarios

- Correctness of scenario must be established
- Additional tooling development

Exceptions to using Model as Oracle

- Non-modeled behaviors
- Low-level hardware mechanism (sensors, telemetry, A/D, etc)
- Exceptional failure conditions the test system cannot expose

Formal Verification: Establishing the Process

New Process Required

- Testing component creation and release
- Testing environment formalization
- Fully reproducible configuration
- Effectiveness of collaborated team work

Focus on Real Issues

- Identification of test system issues
- Identification of repeating issues from regression to regression

Set up Communication

- Work to be done
- Avoid duplication of work on issues with same root cause
- Known issue communication
- Progress

Formal Verification: Validation of Tools

Boston Scientific

Validation typically happens near the end

- Satisfy regulatory requirement
- Master validation plan
- Tool dependency clearly mapped
- Only validated tools are used in formal verification

Model validation

- Cost is high
- Requires requirements experts
- Mostly manual
  - – Opportunities for automation

Formal Verification: Execution and Reporting

Formal Team Formation

- Training
- Specialized roles
- Qualification of team members

Dry Run

- One or two dry runs before formal execution
- Dry run focus on process effectiveness

Group analysis helpful

- “War room” allows team interaction and knowledge sharing

Monitoring and reporting

- Checking for regression completion
- Checking for analysis completion

Cost and Benefits

Cost of Infrastructure

Cost of Quality

Benefit of Lean Testing

Benefit of Re-Use

Cost of Infrastructure

One-Time Costs

- Specification and development of tools (framework)
- Determining what new processes and procedures are needed
- (re-) Training the team

Maintenance Costs

- Maintenance and update of model/architecture
- Tools updates as architecture is enhanced

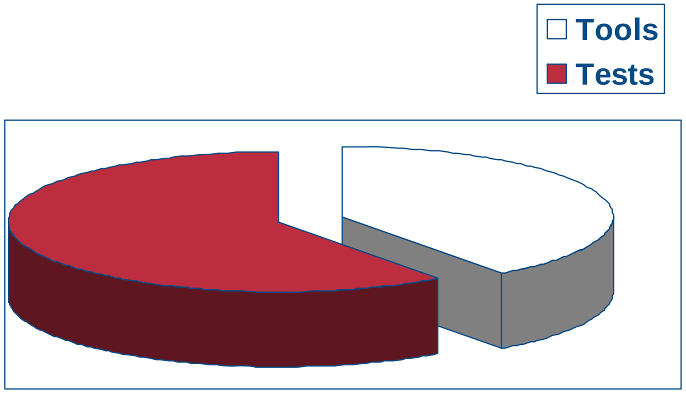
Tools

- Modeling tool may be a financial investment

Cost of Infrastructure

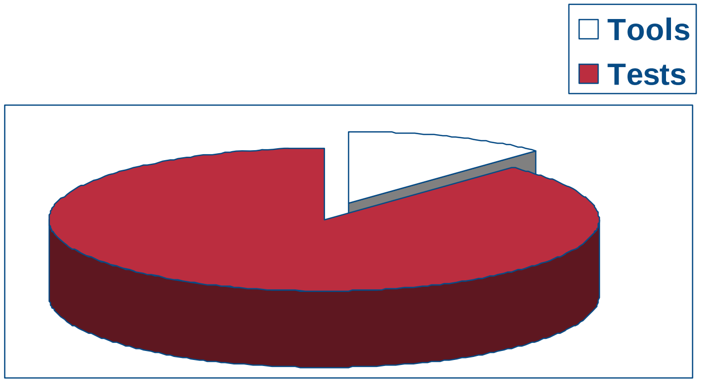
One-Time Costs

40+ %



Maintenance Costs

10-12%



Cost of Quality: Meeting Regulatory Needs

Boston Scientific

Traceability

- Web-based asset management
- Incorporate product line variations

Validation of the Model

- Cross-functional activity

Validation of the Framework Tools

- Dedicated team
- Followed regulatory procedures

Education of regulatory auditors

- New concept may require new submission information

Benefit: Economy of Lean Testing

Boston Scientific

Regression Speed and Frequency

- Fast regression turn around makes regression a less critical factor in regression decision

What-If Testing

- Provide quick response to configuration changes thanks to automated regression process
- Different configuration parameters can be changed, tweaked to suit special testing goals.

Isolating Failures

- Regression rerun with parameter changes allows for easy isolation or confirmation of failures, without test, or target change.

Benefit: Reuse

Boston Scientific

Quality

- We set and achieved a higher standard for test development
- Tests reported more interaction behaviors

Asset reuse

- Test cases are applicable across products
- 90-95% Reuse with no changes
- Assets need change due to requirement change
  - Services, controller etc.

Knowledge capture

- Scenario checkers
- Pattern matchers

Conclusion

Boston Scientific

Original goals:

- Provide support for engineering decision making and communication of decisions
- Improve requirements V & V through the use of the executable requirements model
- Facilitate extensibility and reuse through the use of the reference architecture
- Focus on safety and the ability to incorporate fault tolerant designs

Other beneficial results

- Improved quality of testing
- Capability to do “what if” testing easily
- Greatly improved speed of regression (think “Lean”)

Conclusion

Boston Scientific

Absolute "Musts"

- Focus on really good requirements up front
- Use spiral development
- Leverage the common architecture
- Plan to spend money on tools
- Plan for a dedicated verification debug team on first pass
- Plan for model integration testing