

2003 FAA National Software Conference

Thread Analysis

Thread Analysis: Where the Rubber Meets the Road!



Steve Ward
9/17/03



Overview

- What is it?
- Why do it?
- An approach to selecting requirements.
- Following the requirements down to code – a development view
 - How High-level, Low-Level and Derived requirements relate.
 - How Derived requirements can defeat safety.

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Overview

- Following requirements to test case – a view of verification testing.
- Summary & Questions/Discussion



9/17/2003

What is it?

- A process which follows a single requirement at a time through all the life cycle data.
- The intent is to follow the traceability of a single requirement to ensure a cohesive thread.
 - i.e., requirements => design => code and requirements => test case => procedure => results
- Thread analysis is a sampling technique.
- Job Aid under Item 2.10 in the SOI-2 table notes this technique can be used to evaluate the development and verification data.

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Why Thread?

- Gives one an approach for review of the “technical” Life Cycle data.
- Focuses on the output of the process.
- Samples both development and verification data.
- Good check for systemic errors in the data, especially the test cases.

9/17/2003

Selecting Requirements

- Review the requirements looking for:
 - Safety – need to gain an understanding of how the functions relate to safety,
 - Vague: behavior not defined, terms not specified,
 - Complex logic: multiple conditions and decisions,
 - Anomalous behavior: failure detection and reporting or recovery, and
 - Outputs: emergency/warning/ caution messages, erroneous commands or annunciations.



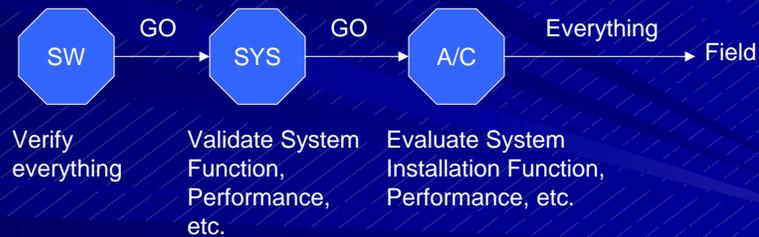
9/17/2003

2003 FAA National Software Conference

Thread Analysis

Selecting Requirements

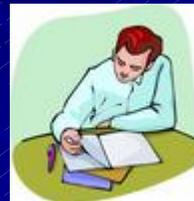
- The GO Path – Everything is working ok!
- If anything is missed during SW verification, it may not get evaluated until fielded.



9/17/2003

Selecting Requirements

- Sometimes different teams develop different functions or areas of the requirements, so pick requirements from different functional areas of the requirements or from different requirement documents.
- Also review the design to help select requirements – don't forget those low-level requirements.
- Look for any areas that may be weak and may not enhance the requirements.
- Look to see if there might be an area that has derived design requirement.



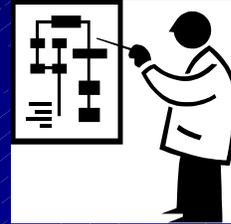
9/17/2003

2003 FAA National Software Conference

Thread Analysis

Going Down the Thread

- Once a requirement is picked, then follow the traceability to the design, code and test case.
- Seek to understand how the traceability was developed – can help in assessing if it was done correctly – was it for before or after development?
- If you are not able to trace out the requirement, continue to thread other requirements – was the trace error part of a systemic problem?
- If the problem was systemic then it needs to be corrected before completing the analysis.



9/17/2003

Going Down the Thread

- In following the requirement down, look at the relationship of the requirements to design and code:
 - Look to see if vague requirements are enhanced,
 - If and how derived low-level design requirements are identified, and if they are ...
 - See if they have been discussed with System Safety.



9/17/2003

2003 FAA National Software Conference

Thread Analysis

Going Down the Thread

- In the code, look to see where each requirement is implemented.
 - Look for code that does not trace up to requirements (i.e., fails a backtrace).
- What the backtrace is looking for is “extra” functionality that could be hidden in the code.
 - i.e., the code has functionality not traceable up to requirements.

9/17/2003

Going Down the Thread

- Backtracing was an audit technique presented at the FAA Job Functions class
- Select a module/program/(whatever you want to call it) and trace each line of code back to a requirement.
- 1st time I used it, I found in-line functionality in the code without any requirements.

9/17/2003

2003 FAA National Software Conference

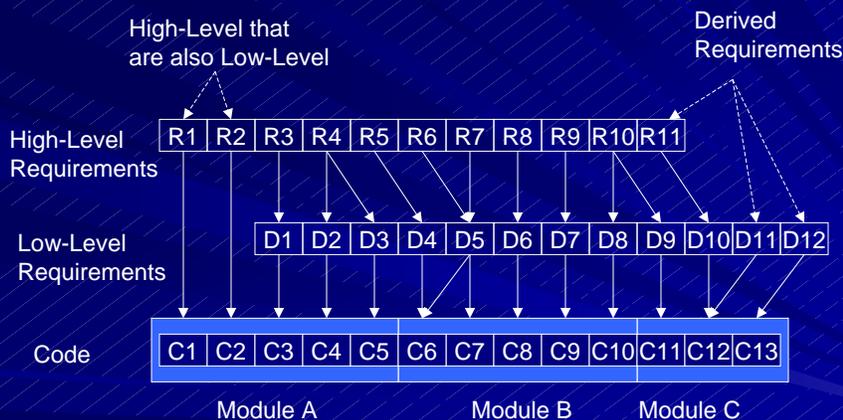
Thread Analysis

A View (but not the only view) of H-L/L-L/Derived

- High-Level defines the function's requirements without regard to design.
- Low-level defines the function's requirements with respect to the design.
- Derived – not directly traceable to higher level requirements.
 - Concern here is “Who is validating these”?

9/17/2003

A Requirement View



9/17/2003

2003 FAA National Software Conference

Thread Analysis

Audio Output Example



- High-Level requirement
 - An audio “Warning” **shall** be output, when the Warning Flag is TRUE
- Low-Level design requirements
 - “Warning” **shall** be written to the audio chip buffer, if the Warning_Flag is TRUE.
 - The audio chip buffer status **shall** be checked for busy before writing to the buffer.

9/17/2003

Audio Output Example

- Code
 - If Warning_Flag = TRUE and Chip_Buffer <> BUSY
 - Chip_Buffer = “Warning”
 - End if (“implicitly” dumping “Warning” on BUSY)

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Audio Output Example

- DO-178B, 5.1.1 SW Requirements Process Objectives and 5.2.1 SW Design Process Objectives state that derived requirements are to be feed back to the systems safety assessment process.
- These derived decisions should only be made by persons with a domain/operational view of the system.

9/17/2003

Audio Output Example

- How the “feedback” to System Safety is to occur needs to be defined by the SW process.
- SW could decide on the appropriate design action and then feed that back to Systems for concurrence.
- However, this might not be the best feedback approach.
- If one leaves the design open with an action to get direction from Systems, then this has better tracking to get closure.

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Audio Output Example

- If “extra” functionality was found in the code by the backtrace, it either needs to be removed or requirements need to be updated to reflect the functionality.
 - Either at the LL or HL requirements.
- Probably the HL should be updated. Not with the chip type detail, but more along the lines of what happens if the SW cannot get the warning message out.

9/17/2003

Audio Output Example

- High-Level requirement
 - An audio “Warning” **shall** be output, when the Warning Flag is True
 - If the audio “Warning” cannot be output in 1 second, fail the function

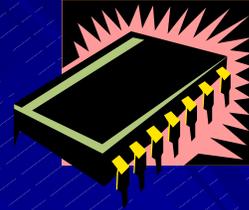
9/17/2003

2003 FAA National Software Conference

Thread Analysis

Audio Output Example

- Low-Level design requirements
 - “Warning” **shall** be written to the audio chip buffer, if the Warning_Flag is TRUE.
 - The audio chip status **shall** be checked for BUSY before writing the audio chip buffer
 - If the audio chip status is BUSY for more than 1 second, set System Fail = True ... (other actions)



9/17/2003

Derived vs Safety

- So, we did our job and developed an “explicit” derived requirement that we needed!
- But... What if we failed to recognize this derived requirement in the design?
- The code might “discover” the buffer status check and just take care of it.
- “Well structural coverage should find it and identify this derived requirement.”
- Maybe not ...



9/17/2003

2003 FAA National Software Conference

Thread Analysis

Derived vs Safety

- Let's say the coder writes the code:
 - IF Warning_Flag= TRUE and Chip_Status <> BUSY
 - Chip_Buffer = "Warning"
 - ENDIF
- Problem: When the functional requirements are verified, the testing may not provide structural coverage of the CHIP_STATUS for Level C and even Level B. Thus, the chip busy status may not get examined if low-level testing is weak.



9/17/2003

Derived vs Safety

- Thus, we cannot leave it to the structural coverage to find derived requirements.
- This could be helped by not allowing the coder to mix H-L and L-L requirements in the same source line.
 - If Warning_Flag = TRUE
 - If Chip_Status <> BUSY
 - Chip_Buffer = "Warning"
 - Endif
 - Endif

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Derived vs Safety

- This would allow hidden requirements to be identified and might help at Level B in having to examine the Chip_Status = BUSY path.
- Might be a good coding standard to define.
- We need to have all the checks/help/etc., to bring any L-L derived design to light.

9/17/2003

Derived vs Safety

- This example shows the importance of testing at both the H-L and L-L.
- This is why the reviews are so important!



- We need to identify and define derived requirements “explicitly”.

9/17/2003

2003 FAA National Software Conference

Thread Analysis

So, We're at the bottom of the Development Thread

- We have examined the requirements, design and code.
- We have looked for derived requirements.
- Did we find any systemic issues?
 - Poor traceability – probably did not get to this point in the thread analysis if so.
 - Implicit derived requirements.
 - Hidden functionality in the code.



9/17/2003

Requirements to Test Case

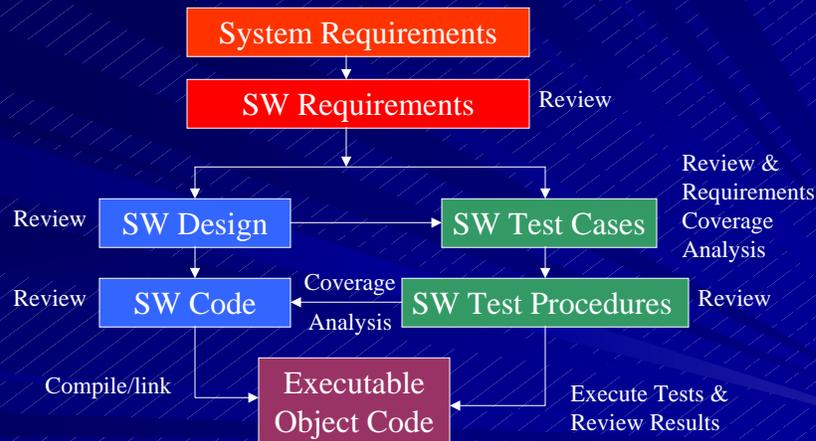
- Note that this is tracing requirement to test case, **not** test procedure.
- This is because the problems with testing mainly lie with test development – understanding of the requirement – not the “coding” of the test procedure.

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Verification Test – a View, but not ...



9/17/2003

Verification Test – a View

- Test Cases are really the “design” for your procedures.
- Many teams tend to skip this step and just move to writing procedures.
- If they do not develop test cases, the chance of having findings goes up!
- Test cases for a requirement should provide a place for the entire test approach to be viewed.

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Verification Test – a View

- I cannot tell you how to assess testing per se, but here are some things to watch for:
 - Look for robustness issues. Some teams try use structural coverage to assess test completeness and miss robustness cases.
 - If you have Level A and complex logic to evaluate, I recommend the MCDC Tutorial on the FAA SW Web page. Again still have to watch the robustness issue.

<http://av-nfo.faa.gov/software/Reports.htm> (NASA/TM-2001-210876)

9/17/2003

Verification Test – a View

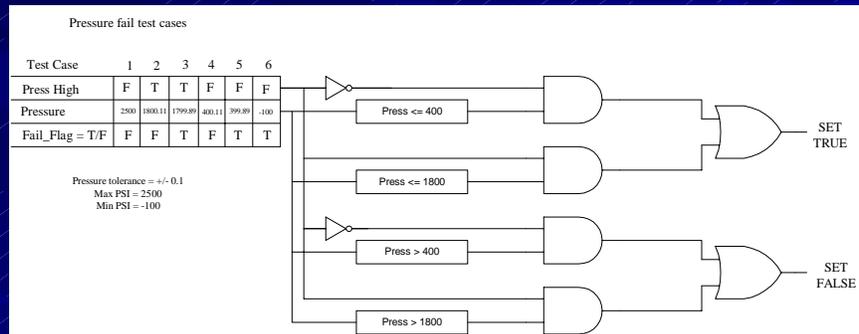
- If there are no test cases, make up your own table of inputs. See if yours agrees with their procedures.
- Let's look at an example:
 - If the High_Press_Flag = True AND the Pressure \leq 1800psi then Fail_Flag = True
 - If the High_Press_Flag = False AND the Pressure \leq 400psi then Fail_Flag = True
 - For any other pressure condition the Fail_Flag = False

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Verification Test – a View



9/17/2003

Verification Test – a View

Pressure fail test cases

Test Case	1	2	3	4	5	6
Press High	F	T	T	F	F	F
Pressure	2500	1800.11	1799.89	400.11	399.89	-100
Fail_Flag = T/F	F	F	T	F	T	T

Pressure tolerance = +/- 0.1
Max PSI = 2500
Min PSI = -100

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Verification Test – a View

- If you don't understand the test or think you have found a problem,
 - Ask the engineer to explain it
 - Don't put the engineer on the defensive
 - Sometimes this leads to their finding an issue
 - Find another DER/expert to help review

9/17/2003

Requirements to Test Case

- Once you are satisfied the Test Case is correct then on to the Test Procedure.
- Need to make sure it “implements” the Test Case correctly.
- And only the Test Case!
 - Have found the procedure doing more than it should - extra tests to cover conditions within a decision
 - Can cause incorrect coverage analysis results.

9/17/2003

2003 FAA National Software Conference

Thread Analysis

Requirements to Test Case

- Last we need to look at the Test Results.
 - Is the expected result clearly shown?
 - Is the pass/fail criteria clearly shown and met?
 - Are any failures explained or is there a problem report written?



9/17/2003

Requirements to Test Case

- Well I thought we would never get to the bottom of the verification test thread.
- Again did we find any systemic issues?
 - Poor traceability?
 - Weak Test Case(s)?
 - Procedures too “robust”?
 - Inadequate test results?
 - Failures not resolved?



9/17/2003

2003 FAA National Software Conference

Thread Analysis

Summary

- Thought needs to go into the selection of requirements to thread
- Threading is a good approach for:
 - reviewing HL-LL and identifying possible derived requirements,
 - accessing technical data,
 - finding systemic issue in verification test.



9/17/2003

Summary

- Derived requirements must be identified to Systems Safety for consideration of safety effects.
- Process is important – reviews must be able to identify derived requirements.
- Sampling of testing adequacy is needed to identify systemic testing issues
- It helps focus the review and the reviewer's understanding of the software.



9/17/2003

2003 FAA National Software Conference

Thread Analysis

