U.S. Department of Health
and Human Services

Center for Devices and
Radiological Health

MODEL BASED ENGINEERING: Software Quality Metrics & Assurance Cases

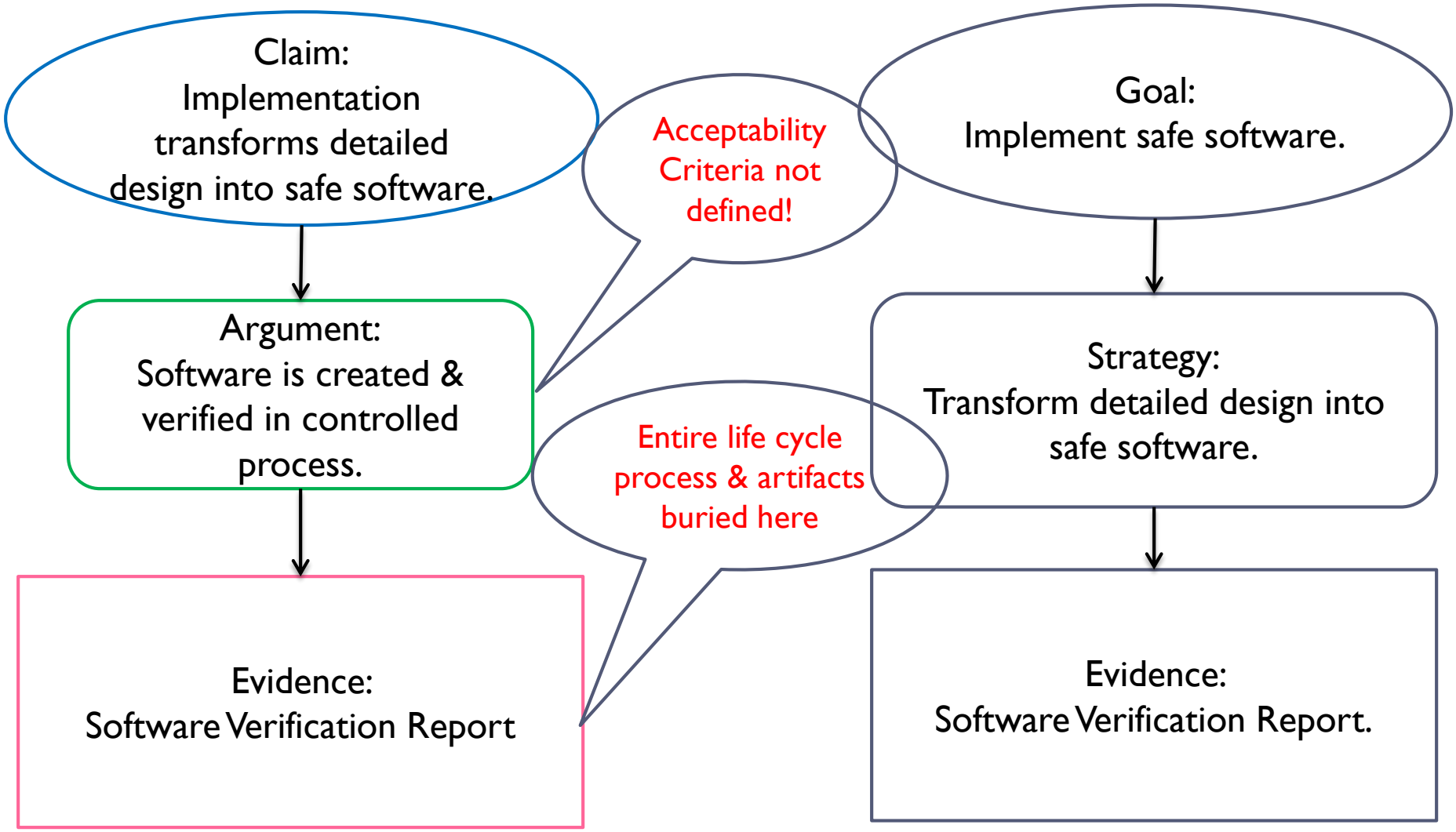
Paul Jones

Senior Systems / Software Engineer

Division of Imaging, Diagnostics, and Software Reliability

Center for Devices and Radiological Health

Motivation Examples



Purpose

- To see what software quality metrics are possible in a Model Based Design/Engineering process.
- To explore how software quality metrics and software quality measurements can be used as arguments and evidence in assurance cases.

Caveat

Recognize that there will likely be some delta between a model and reality.

Assumptions

- A quality system life cycle process is in place
- A software life cycle quality system process is in place
- A safety life cycle quality system process is in place
- A security life cycle quality system process is in place

Terminology

- Acceptable: Able to be tolerated or allowed [1]
- Criteria: A principle or standard by which something may be judged or decided[1]
- Acceptability criteria[1] :

  A principle or standard by which something (risk, design requirements, verification / validation results, etc.) may be judged or decided.
- Consistent[2] :

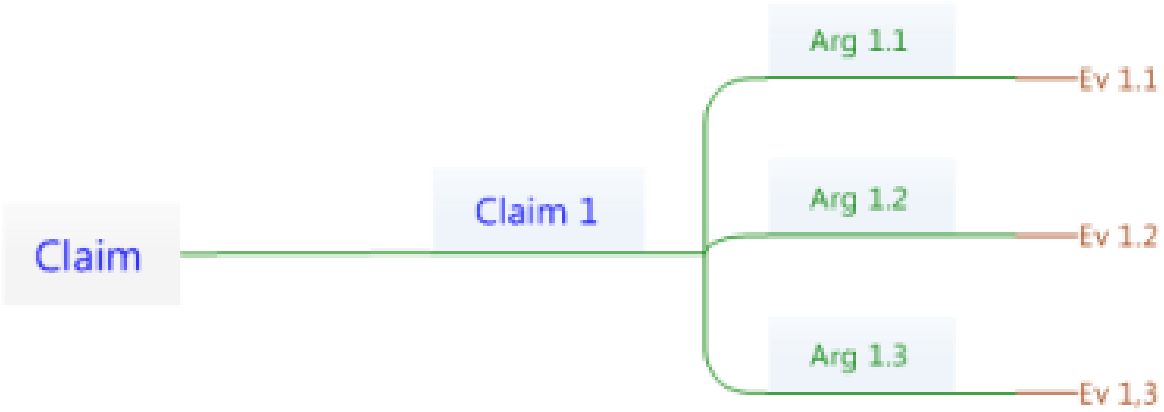
  The requirement does not contradict any other requirement and is fully consistent with all authoritative external documentation (including model).
- Complete[2] : The requirement is fully stated in one place with no missing information.
- Unambiguous[2] :

  It expresses objective facts, not subjective opinions. It is subject to one and only one interpretation.
- Verfiable[2]

  The implementation of the requirement can be determined through basic possible methods: inspection, demonstration, test (instrumented) or analysis (to include validated modeling & simulation).

1. https://www.google.com/search?q=acceptable+definition&ie=utf-8&oe=utf-8
2. https://en.wikipedia.org/wiki/Requirement

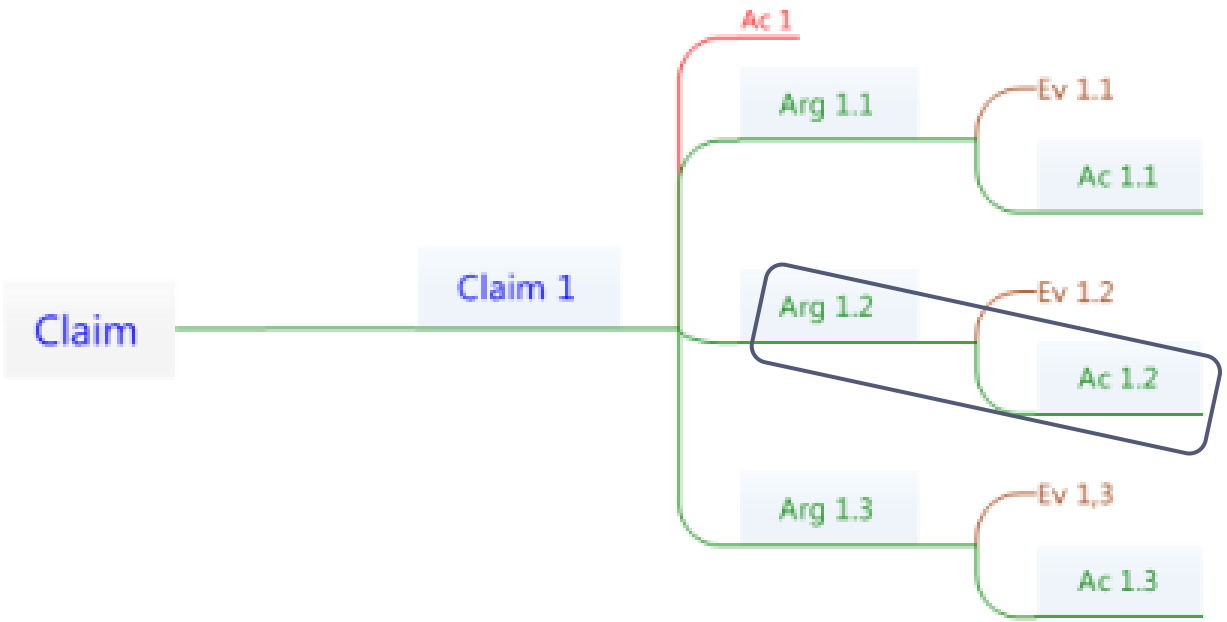
Atomic Assurance Case Tuple (C,A,E)



Arg – Argument
Ev - Evidence

Argument uses **E**vidence to *Justify* **C**laim

Assurance Case Argument Pair (Arg, Ac)



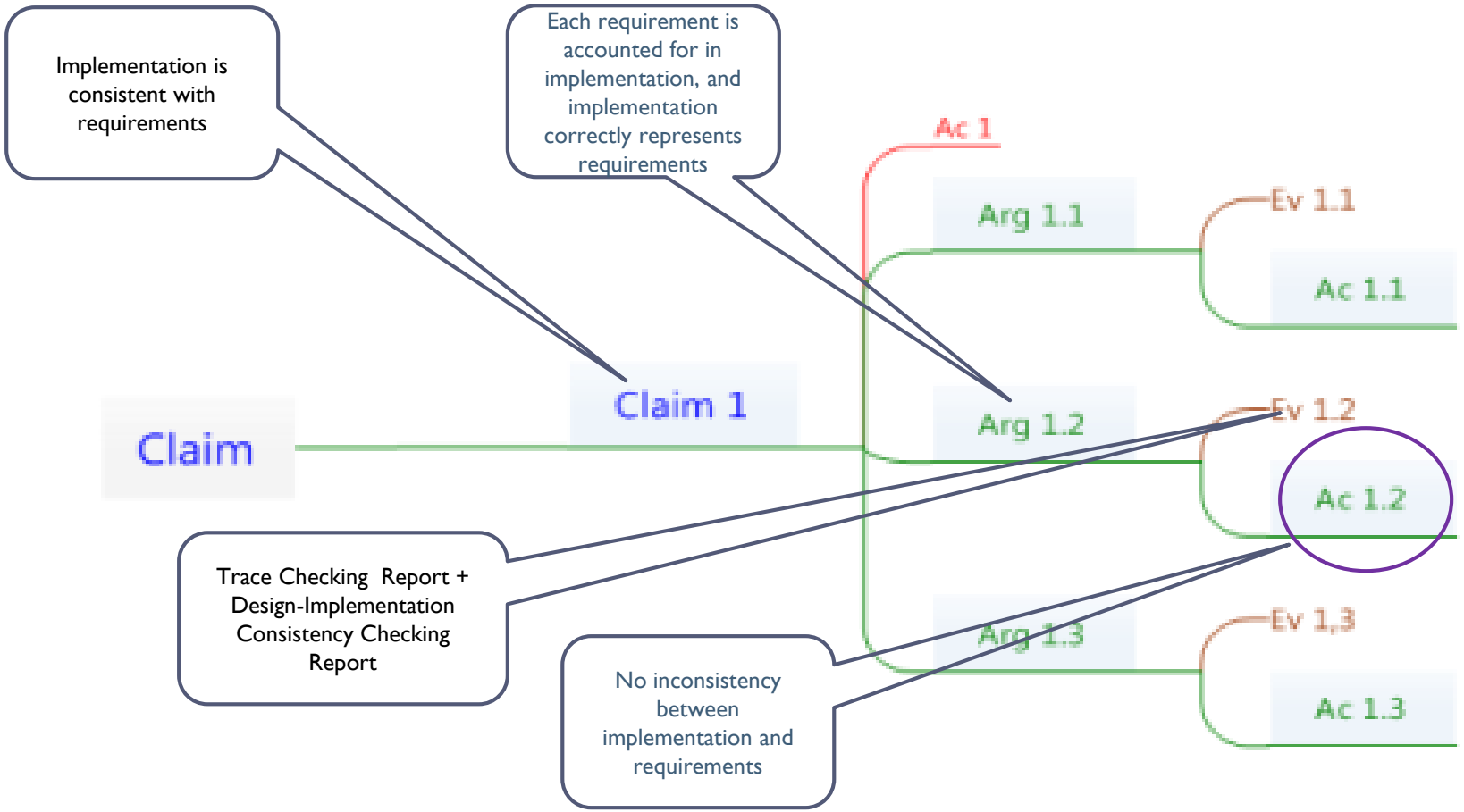
Arg – Argument
Ev - Evidence
Ac - Acceptability Criteria

Arg -> means, manner, method or logic that uses Evidence to justify Claim 1
Ac -> "measure" that refers to Evidence to substantiate Argument
NOTE: Ac1 can be NULL if Ac1.1, Ac1.2, and Ac1.3 substantiate Claim 1

Measures / Metrics

Acceptability Criteria establishes a basis for measuring or judging whether or not something has been acceptably achieved.

Example from Software Domain



Model Based Design Process

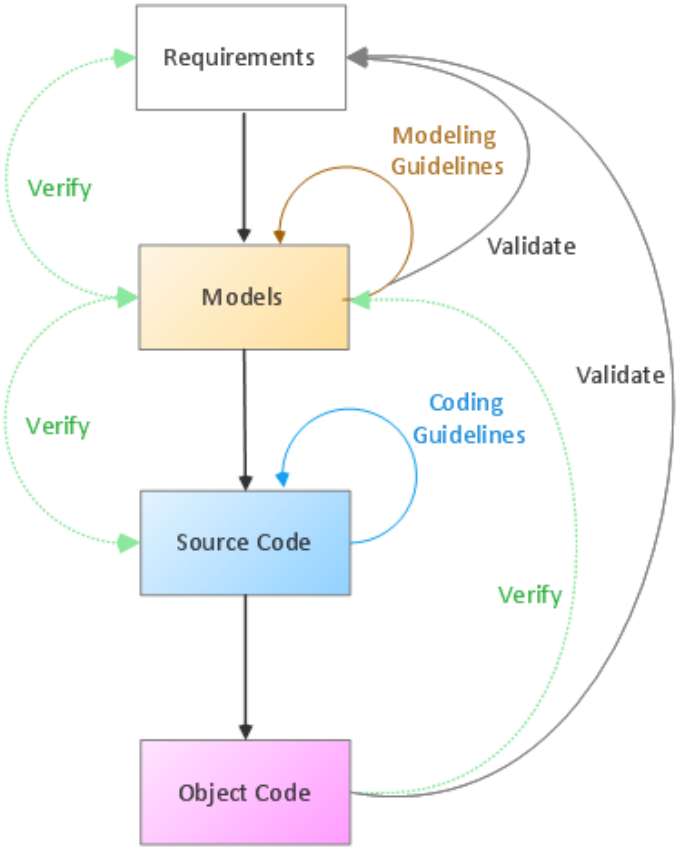


Image taken and enhanced from Mathworks with permission by Dave Hoadley

MBD Process Tool Chains

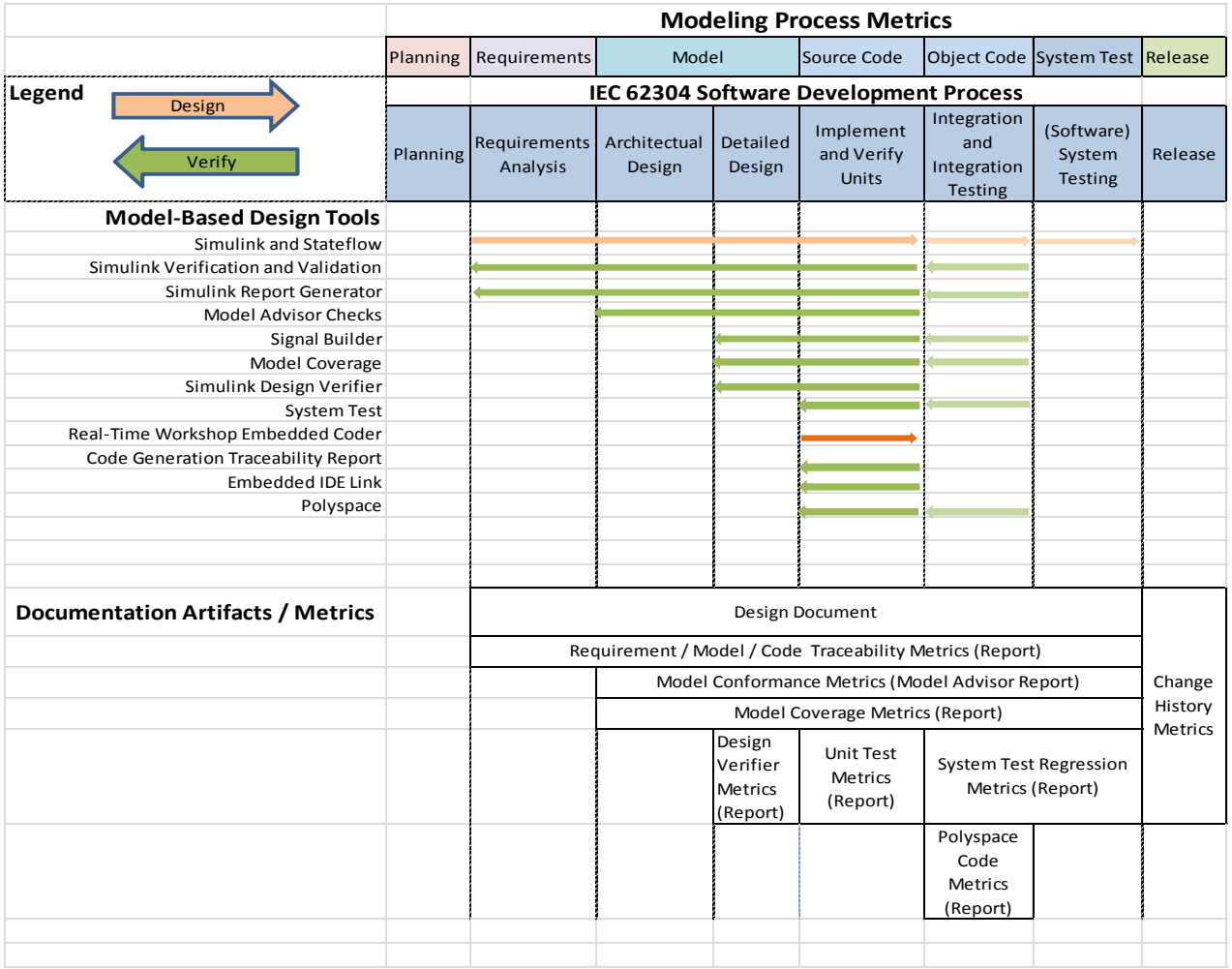
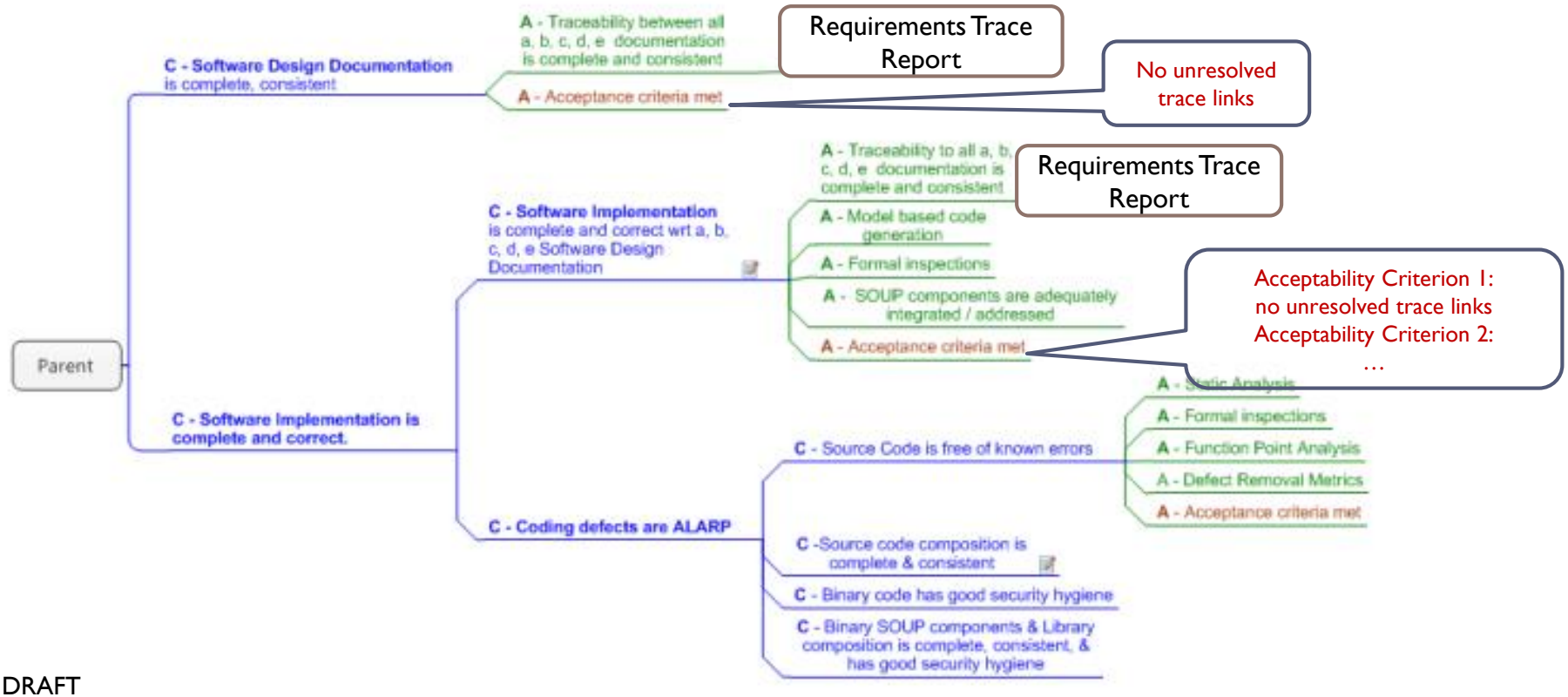


Image taken from Mathworks and modified with permission by Dave Hoadley (FDA does not endorse Mathworks products)

Requirements / Model / Code Traceability Report

- Identifies links between:
  - Natural Language Requirements → Requirements
  - Requirements → Model Architecture Constructs → Model Blocks → Code Units
  - Requirements → Test Cases
  - Code Units → Test Cases
- Identifies dangling and unaccounted links, e.g.:
  - Identifies Model Blocks for which there are no links to Requirements
  - Identifies Requirements for which there are no links to Model Blocks (i.e. dangling Requirements)

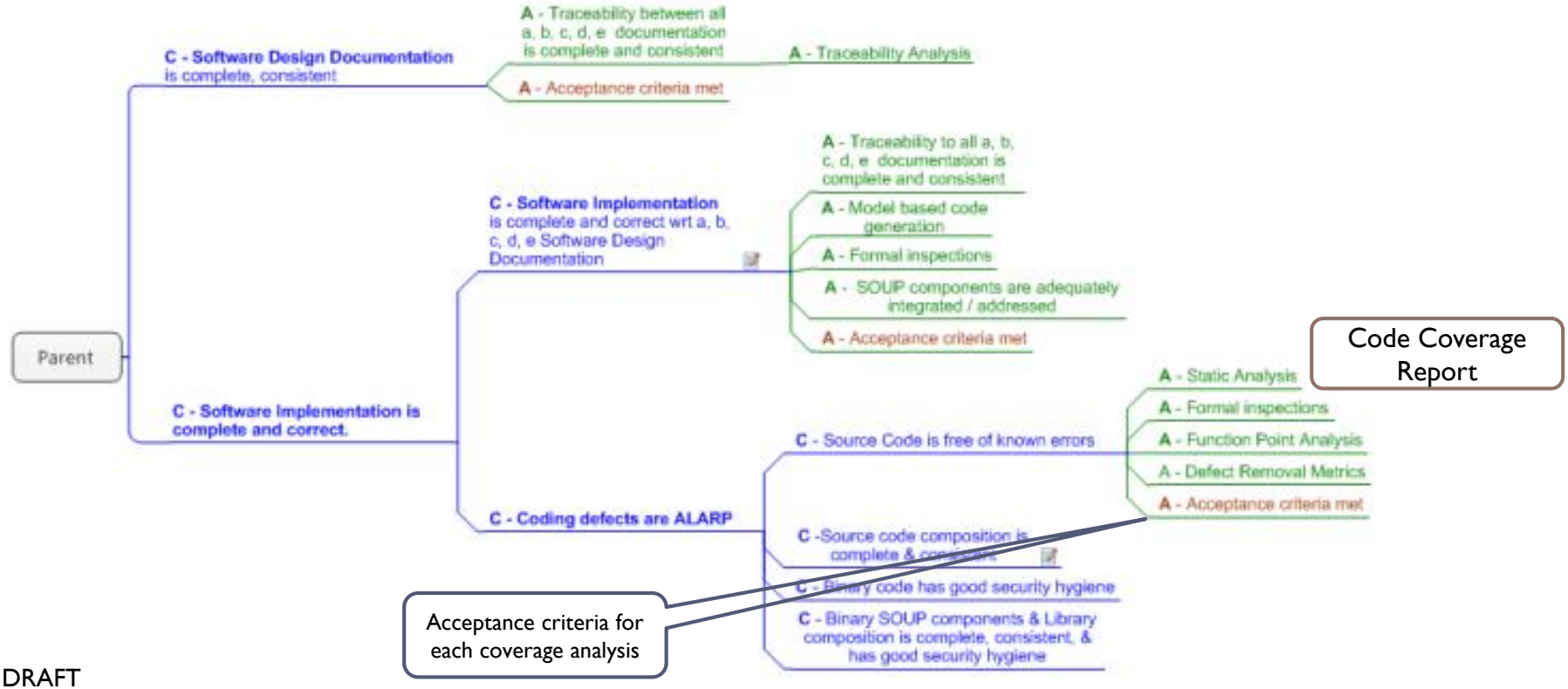
Possible Software Quality Metrics-Based Assurance Case



DRAFT

Software Design/Implementation Trace assurance fragment

Possible Software Quality Metrics-Based Assurance Case



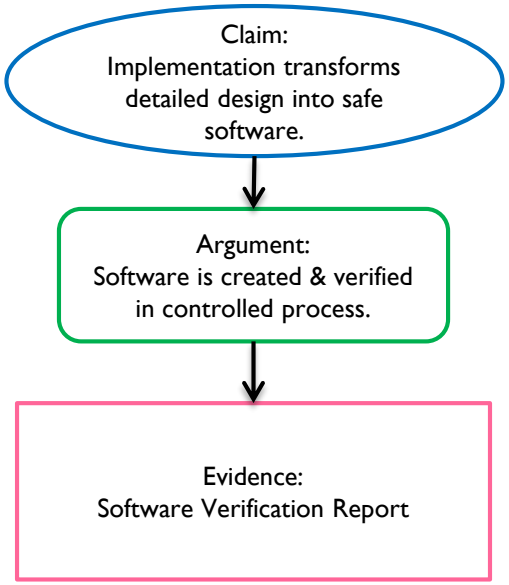
Software Design / Implementation Trace assurance fragment

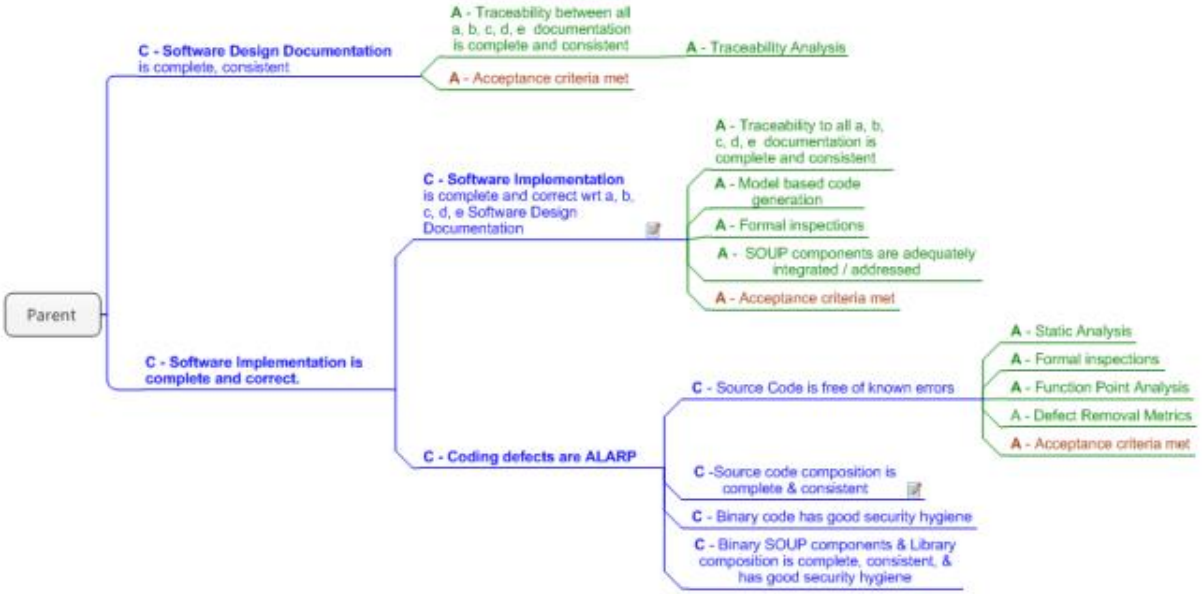
Code Coverage Reports

Run-time errors, concurrency issues, security vulnerabilities, and other defects in C and C++ embedded software using static analysis.

- Cyclomatic complexity coverage
- Condition coverage
- Decision coverage
- Modified condition/decision (MCDC) coverage
- Saturate on integer overflow coverage
- Relational boundary coverage
- Signal range coverage
- Signal size coverage
- Data Flow Checks
- Numerical Checks
- Static Memory Checks
- Control Flow Checks
- Type Check

Current vs (Possible) Future Software Assurance Case





Current

(Possible) Future

Research

1. What are the Quality Metrics for software?
2. For each quality metric, is it practical to establish consensus on Acceptability Criteria among stakeholders?
3. What is the (Arg, Ac) pair stopping criteria?

   i.e. when is an argument justifying the Acceptance Criteria unnecessary?
4. Do software Quality Metrics and corresponding Acceptability Criteria contribute to confidence?
   - If so, can this confidence be measured in some uniform, objective, and/or quantitative manner?

Thank You!