Raytheon
Customer Success Is Our Mission

Algorithm Description Documentation and Validation Process

8th Annual Systems Engineering Conference, National Defense Industrial Association,
San Diego, CA

Mike Bailey
Raytheon Company
714.446.4195
mkbailey@raytheon.com

10/26/05

Abstract

- Complicated algorithms difficult to translate to SRS
  - Highlights the division between algorithm validation and software verification
  - Results in a disjoint agreement between Systems and Software engineering
- Algorithm Description Document
  - Documents the life-cycle of algorithms
  - Includes trade study analyses and validation results to illustrate details design intent
  - Allows for a mutual engineering understanding
  - Describes the most recent design
- ADD Process is presented

ADD defines the difference between Verification and Validation

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

Navigation and Landing Systems Developments

- Raytheon’s NLS group developed ADDs and simulation tools to test algorithms
  - Process successfully implemented during initial operating phase of an SBAS approved for use in commercial aviation
  - Customer and Raytheon currently planning upgrades to improve service and availability
  - A combination of algorithm enhancements required to achieve future operational goals
    - □ To be rolled out sequentially over the next few years
- Process expanded to include all safety algorithms in NLS program subsystems
  - Continues to be used by all SBAS programs within Raytheon’s NLS group as the primary algorithm development tool

ADD allows Software Enhancements

Software Development Challenges

- Typical programs include safety-of-life systems centered on math-intensive algorithms
  - New insights in antenna design, ionospheric behavior and error mitigation lead to algorithm redesigns
- The safety-of-life requirement suggests the use of an RTC/DO-178B Level B process
- NLS programs consist of software developed to DO-178B Level B and Level D standards
  - Level B software passes through rigorous set of design and testing requirements
  - Level B coding involves creating a formal SRS, ensuring that all requirements in the SRS are addressed in the code, and formally testing all branches of the code for conformity

Ensure software meets safety requirements

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

Software Requirements Document Considerations

Raytheon

- SRS is relatively expensive
  - Each idea broken down into modules used to generate pseudo code
  - Common for SRS updates to lag
- SRS focuses on how code is supposed to operate
  - Does not capture discussions and trades that justify the algorithms
  - Various filters and algorithms in the system all require analysis
- Scientists operate in a results-based paradigm
  - More effort is concentrated on the results of the algorithms to be correct than that pseudo code to be clear
  - As a consequence, system prototypes were correct implementations of the SRS but not correct implementations of the algorithms envisioned
  - Software may have been appropriately verified, but was not a guaranteed implementation of validated algorithm

Methodology concentrates on Algorithm Validation

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

System Engineering Methodology

- Design, prototype, tune and validate
- Process has 3 goals
  - Allow flexibility to rework algorithms and code
  - Capture information that describes how decisions were made
  - Focus on ideas and results, not implementation details

Algorithms are proposed and described in ADD

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

Algorithm Generation

- Offline studies
  - – Trading one approach against another or examining historical data
  - – Included in the ADD to give an understanding for the motivation of the algorithm

- Design is prototyped into a simulation of deliverable system
  - – If the design has not been fully decided, the prototype engineer will use best judgment to get a version working
  - – If there are multiple competing designs, they are coded under compile flags and comparison simulations are executed

ADD includes off-line studies

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

Algorithm Validation

- To validate that code operates as expected the simulation is executed
  - Code has extensive debugging capabilities to generate reports of algorithm functions
  - Compared to the algorithm description in the ADD to determine if it has the correct behavior
- Tuning effort where the algorithm is optimized
  - Algorithm is modified in the simulation until it meets expectation results
  - Summarized in ADD
- Once tuned, algorithm design described in ADD is updated
- ADD reviewed to ensure that the results are correct and that all anomalies are explained
- ADD given to Software Engineering for implementation

All validation efforts summarized in ADD

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

Software Engineering Methodology

- Generate software requirements, code and verify

- Process has 3 goals
  - Ensure one-to-one correspondence between specification and code
    - No unimplemented requirements and
    - No code that is not described in the requirements
  - Ensure that the software has been coded to meet the prescribed RTC/DO-178B safety level
    - All branches tested for correctness and robustness

Methodology concentrates on algorithm validation

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

Algorithm Implementation

- Software Engineering updates the SRS with requirements generated from ADD
- Design review held to ensure the algorithm described in the ADD can be implemented in a manner consistent with Level B design
  - It must be possible to prove that the code will operate consistently in a manner as described in the SRS
  - System operational software is coded from the requirements
    - □ If the deliverable code is being generated by perfecting simulation code, the Software Engineer must ensure that the SRS, and not the prototype, is used as the source of requirements.
  - The operational code is to be free of dead or unused code, debugging code and any version of the algorithm other than the final version described in the SRS

SRS updated from ADD

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

RTC/DO-178B Level B Coding

- Level B software is coded in a style that prohibits unsafe programming constructions
- Software Engineers specially trained in Level B programming techniques
- Once complete, the simulation code is available as a resource for comparison testing
- Discrepancy between deliverable and prototype code is comparatively great
  - Prototype coders are engineers and mathematicians focused on validation and benefit exclusively from executing the simulation
  - Prototype not developed to meet any specific coding standards

Operational code required to pass though rigorous design and testing requirements

Limitations

- There is risk in overestimating the code correctness of the validated algorithms in the simulation
  - The simulation is validated by checking that its algorithms function as expected
  - It is possible to have simulation code that gives correct or nearly correct results while still containing coding errors

- If integration-level testing of the operational software is less rigorous because of confidence in the algorithm validation of simulation code, some corner robust cases may be under tested

Robust cases may be under tested

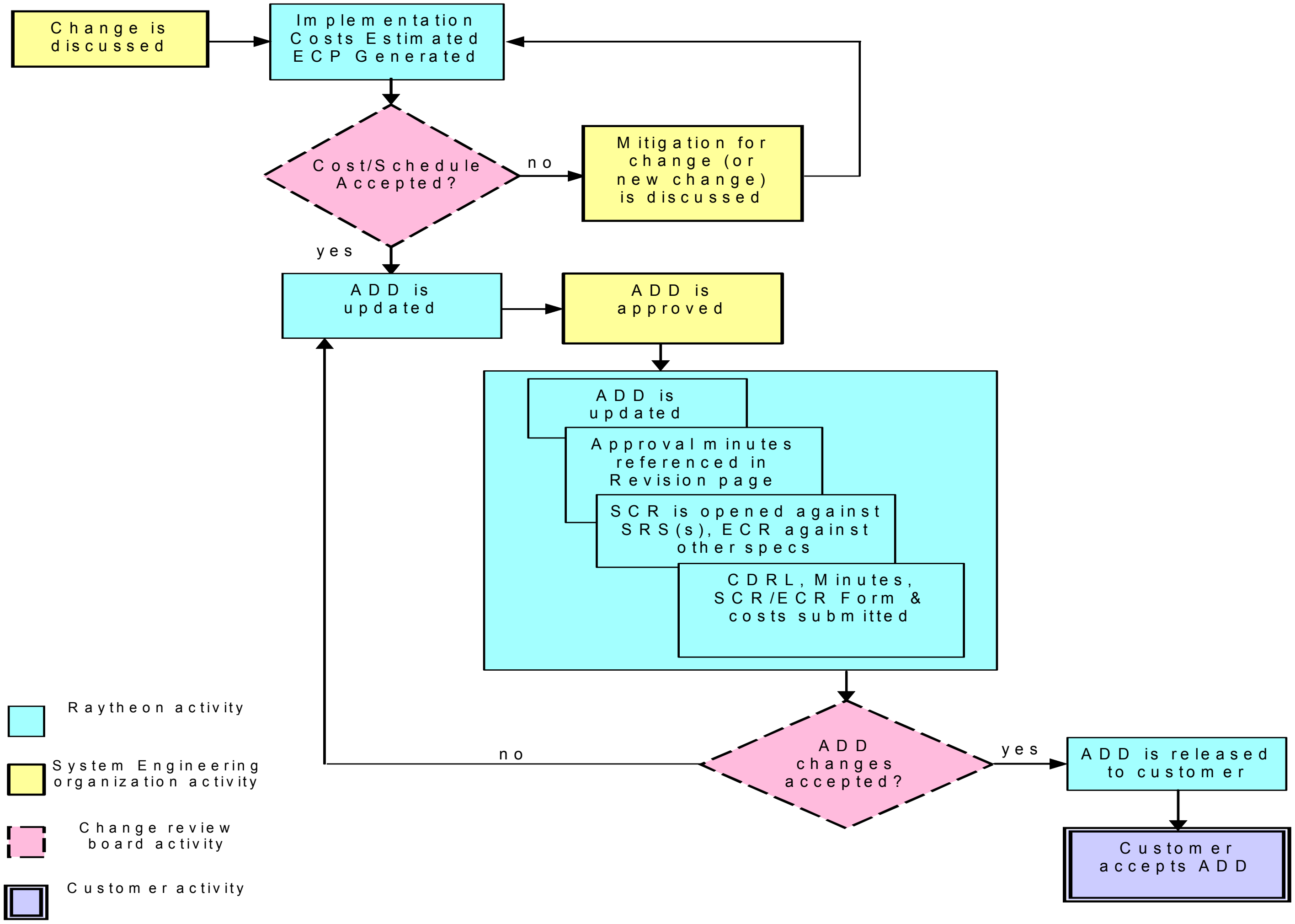
The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

Algorithm Changes

- ADD under control of Algorithm Design Team
- SRS under control of Software Engineers
  - ADD and SRS kept synchronized to facilitate communication between the two groups
  - In this way a Software Engineer is unable to originate a change to an algorithm that is not reviewed by the algorithm designer
  - Similarly, the analyst is unable to introduce a subtle change in the algorithm that is not captured in the code
- Once the algorithms themselves are defined, the algorithms and related information are documented in the

ADD is approved by System Engineering

Change Process


Change is discussed
Implementation Costs Estimated ECP Generated
Cost/Schedule Accepted?
no
Mitigation for change (or new change) is discussed
yes
ADD is updated
ADD is approved
ADD is updated
Approval minutes referenced in Revision page
SCR is opened against SRS(s), ECR against other specs
CDRL, Minutes, SCR/ECR Form & costs submitted
ADD changes accepted?
no
yes
ADD is released to customer
Customer accepts ADD
Raytheon activity
System Engineering organization activity
Change review board activity
Customer activity

Changes are made and approved by a change review board

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.

Conclusion

- ADDs and a validation process has helped programs to be more agile in the face of rapid algorithm redesign
- By centralizing information on algorithm tuning and validation, it is easier to understand the history and justification of decisions made
- By formatting the information to be more accessible to the engineers and scientists, it has kept them more engaged in the process of document review

ADDs have led to a safer and more correct product

The contents of this material reflect the views of the authors. Neither the Federal Aviation Administration nor the Department of Transportation make any warranty or guarantee, or promise, expressed or implied, concerning the content or accuracy of the views expressed herein.