
Software

Change Impact Analysis

An Interactive Video Teletraining Course

IVT Course 62823

Self-Study Video Course 25823



Developed and Presented by
Leanna Rierson
FAA, National Resource Specialist
For Aircraft Computer Software

Aircraft Certification Service
Federal Aviation Administration
May 11, 2000

CONTENTS OF IVT GUIDE

How Do I Use This IVT Guide?

IVT Course Orientation

What Is IVT?

Who Is the Target Audience?

Who Is the Instructor?

What Will You Learn?

What Does the Lesson Cover?

Appendices

- A. Presentation Visuals
- B. N8110.M&M – Software Change Impact Analysis Notice
- C. Draft Paper – Planning for Software Changes
- D. Example CIA – AlliedSignal TCAS
- E. Example CIA – Honeywell Lateral Guidance Software
- F. Example Procedure - AlliedSignal
- G. Example Procedure – Honeywell Defense
- H. Example Process Presentation – Honeywell Phoenix
- I. Evaluation Forms

How Do I Use This IVT Guide?

This Interactive Video Teletraining (IVT) Guide provides you with an orientation to the IVT presentation, support materials for use during the broadcast, and the course evaluation.

Follow these steps to complete your study.

1. Review the *IVT Presentation Orientation* before the broadcast, if possible, or before you watch the self-study videotape. It provides the purpose of the presentation, the target audience, information about the instructor, what you will learn, and topics covered.
2. Turn to Appendix A, *IVT Presentation Visuals*, and refer to it during the broadcast/videotape. You can use these visuals to take notes and follow along when viewing the presentation/ self-study video.
3. Review the software notice and paper in Appendices B-C before the broadcast, if possible, or before you watch the self-study videotape.
4. If completing this course by watching the video, please complete the *Self-Study Video Evaluation Form* in Appendix I and send it to your Directorate/Division Training Manager (ATM). Your comments are very important to us and will help to enhance the quality of the IVT lesson. If you are participating in the live IVT broadcast, you will obtain instructions to complete your evaluation at the end of the broadcast, using the *IVT Course Evaluation Form* and your keypads.

NOTE: The IVT broadcast will be videotaped so that it may be used as a self-study package for those who were unable to participate in the broadcast, or for those who wish to refresh their knowledge of the content presented. This IVT Guide may also be used with the self-study videotape.

What Is IVT?

Interactive Video Teletraining, or IVT, is instruction delivered using some form of live, interactive television. This course originates from the television studio at the FAA Academy in Oklahoma City. Through the IVT broadcast facility, the instructor is able to use a variety of visuals, objects, and media formats to support the instruction.

Participants are located at various receive sites around the country and can see the instructor and his/her materials on television sets in their classrooms. The participants can communicate with the instructor either through a microphone and/or the simple-to-use Viewer Response System keypads. During the live presentation, when a participant has a question or the instructor asks for specific participant responses to questions, the participant(s) can signal to the instructor using the keypad.

The collective participant responses, or the name of a specific participant signaling a question, are immediately visible to the instructor on the console at the broadcast site. The instructor can then respond as needed. When the instructor calls on a specific participant to speak from a site, participants at each of the other sites can simultaneously hear the participant who is speaking.

This guide provides you with a framework for this course as well as the following three appendices to be used during the course:

- Appendix A contains copies of the actual slides used by the instructor during the broadcast. You can use these visuals to follow along with the broadcast or when you watch the tape and to record notes directly on the pages.
- Appendices B-H contain the documents that will be discussed throughout the broadcast.
- Appendix I contains the Course Evaluation Forms. You will be given instructions on how to complete the evaluation during the broadcast. If you are watching the video, upon completion, please fill out the *Self Study Video Evaluation Form* and send to your Directorate/Division Training Manager (ATM). If you do not send the form to your ATM, you will not be able to receive credit for this course.

Who Is the Target Audience?

Engineers who are responsible for approving software.

Who Is the Instructor?

Leanna Rierson is the National Resource Specialist for Aircraft Computer Software. She has 12 years of experience in the computer/aviation industry. These positions include: national software program manager of the FAA Avionics Branch (AIR-130), avionics/electrical engineering specialist at the Wichita ACO, and software positions with industry at NCR and Cessna Aircraft Company. Leanna graduated summa cum laude from Wichita State University, has a Master's degree in Software Engineering, and is currently pursuing a PhD. Leanna leads numerous efforts, including the following: FAA's Software Grand Design team, FAA's Streamlining Software Aspects of Certification program, international Certification Authorities Software team, RTCA Special Committee #190 editorial team, FAA's Technical ReUsable Software Team, and the Software and Digital Systems Research Technical Community.

What Will You Learn?

At the end of the training, participants will be able to:

- Explain the importance of good software change process.
- Describe the current policy on change impact analysis.
- Explain the importance of regression testing.
- Describe the FAA policy on the major/minor change classification process for software changes.

Self-Assessment

Pre- & Post-Course Self-Assessment Questions

If you are taking this course via IVT and you are logged on to a keypad, you will be asked before and after the broadcast to complete this self assessment, using your keypads. If you are taking this via self-study video, please complete manually and return with your end of course evaluation to your directorate/division training manager (ATM).

Rate your confidence level for each of the following statements before and after completing the course.

1. I can explain the purpose of software policy.

	<u>Very Confident</u>	<u>Moderately Confident</u>	<u>Not Confident</u>
BEFORE THE COURSE:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AFTER THE COURSE:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

2. I can describe the current policy on change impact analysis?

	<u>Very Confident</u>	<u>Moderately Confident</u>	<u>Not Confident</u>
BEFORE THE COURSE:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AFTER THE COURSE:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

3. I can explain the software change process that an applicant follows?

	<u>Very Confident</u>	<u>Moderately Confident</u>	<u>Not Confident</u>
BEFORE THE COURSE:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AFTER THE COURSE:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

4. I can describe approval the process of evaluation major/minor change classification procedures.

	<u>Very Confident</u>	<u>Moderately Confident</u>	<u>Not Confident</u>
BEFORE THE COURSE:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
AFTER THE COURSE:	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Presentation Visuals

Appendix A

Software Change Impact Analysis



May 2000

Leanna Rierson

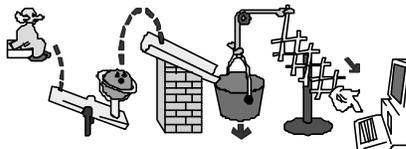
**National Resource Specialist
for Aircraft Computer Software**

IVT Hotline: (888) 279-8604

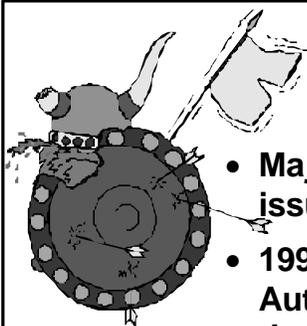
1

Course Objectives

- Explain the importance of an effective software change process.
- Describe the current FAA policy on:
 - Software change impact analysis.
 - Major/minor change classification process for software changes.
- Explain the importance of verifying software changes.



2



History (1/2)

- **Major/Minor change classification issues have existed for years.**
- **1993-1996 - Certification Authorities Software Team (CAST) developed a position paper on major/minor software classification.**
- **1998 - Streamlining Software Aspects of Certification (SSAC) Industry Workshop participants raised issue regarding FAA policy on major/minor software changes.**

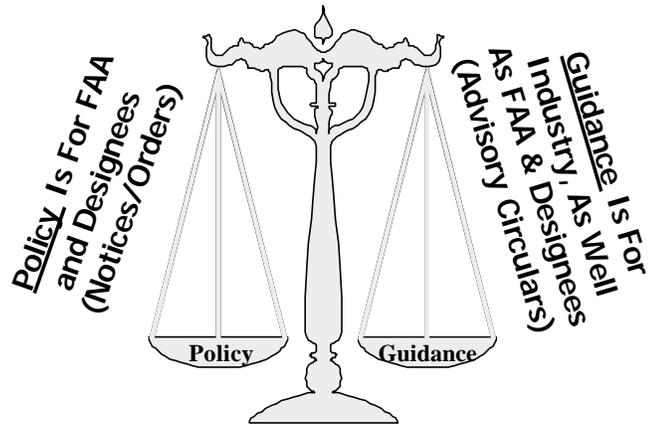
3

History (2/2)

- **1998-1999 - FAA and industry worked together to develop a position addressing major/minor classification of software changes.**
- **Result: Software change impact analysis guidelines.**
- **Policy: Notice entitled “Guidelines for the Oversight of Software Change Impact Analysis Used to Classify Software Changes As Major or Minor.”**
- **Status: Notice forwarded for signature December, 1999.**

4

Policy vs. Guidance



5

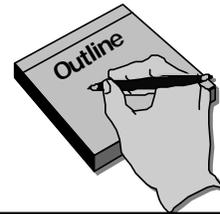
Developing A Notice



6

Notice N8110.M&M Outline

- Title: “Guidelines for the Oversight of Software Change Impact Analysis Used to Classify Software Changes As Major or Minor”
- **Section 1: Purpose**
- **Section 2: Distribution**
- **Section 3: Related Publications**
- **Section 4: Background**
- **Section 5: Discussion**
- **Section 6: Procedures**
- **Section 7: Conclusion**

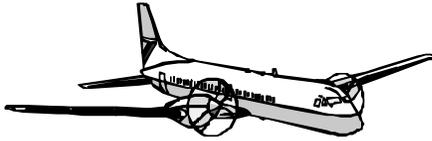


7

Section 1-3: Purpose, Distribution, Related Publications

- **Section 1 - Purpose**
 - To provide a standardized process for assessing the safety impact of software changes & determining FAA’s involvement in software changes
- **Section 2 - Distribution**
 - FAA and designees
- **Section 3 - Related Publications**
 - Advisory Circular 20-115B
 - RTCA/DO-178B
 - Part 21
- **Note:** Notice N8110.78 on Legacy Systems is also related

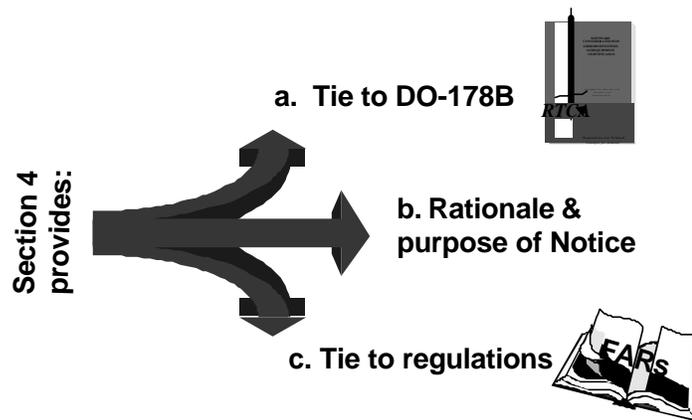
8



Section 4 - Background

9

Overview of Section 4



10

Tie to DO-178B

- AC 20-115B recognizes DO-178B as a means of compliance to the FARs
- DO-178B addresses software changes in Section 12.1



11

Tie to Regulations

- Regulations address major/minor changes in:
 - 21.93(a) and 21.95 - Type Certification
 - 21.611(a), (b) - Technical Standard Order (TSO)
- Regulations look at changes from the product perspective



12

FAR Quotes (1/2)



- 21.93(a) states that a “‘minor change’ is one that has no appreciable effect on the weight, balance, structural strength, reliability, operational characteristics, or other characteristics affecting the airworthiness of the product. All other changes are ‘major changes’ ...”.
- 21.95 states: “Minor changes in a type design may be approved under a method acceptable to the Administrator before submitting to the Administrator any substantiating or descriptive data.”

13

FAR Quotes (2/2)

- 21.611 (a) and (b) addresses “minor” and “major” changes for TSO manufacturers. 21.611(a) basically says that minor changes (i.e., a change that’s not major) may be made without further approval by the FAA. The revised data should be submitted to the appropriate ACO. 21.611 (b) states that “Any design change by the manufacturer that is extensive enough to require a substantially complete investigation to determine compliance with a TSO is a major change.”

14

Types of Software Changes (1/2)

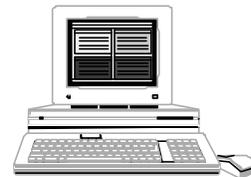
- **Pre-Certification**
 - During software development/before software approval
 - Change control in place
 - Problem reporting & correction in place
 - Re-verification in place
 - Addressed in Sections 7 & 8 of DO-178B



15

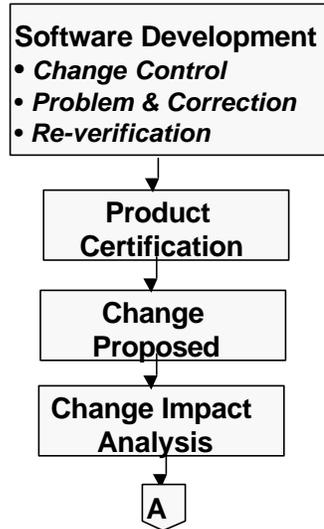
Types of Software Changes (2/2)

- **Post-Certification**
 - After software approval and product certification
 - Section 12.1 of DO-178B (“Use of Previously Developed Software”) addresses this kind of change
 - N 8110.M&M focuses on the post-certification change



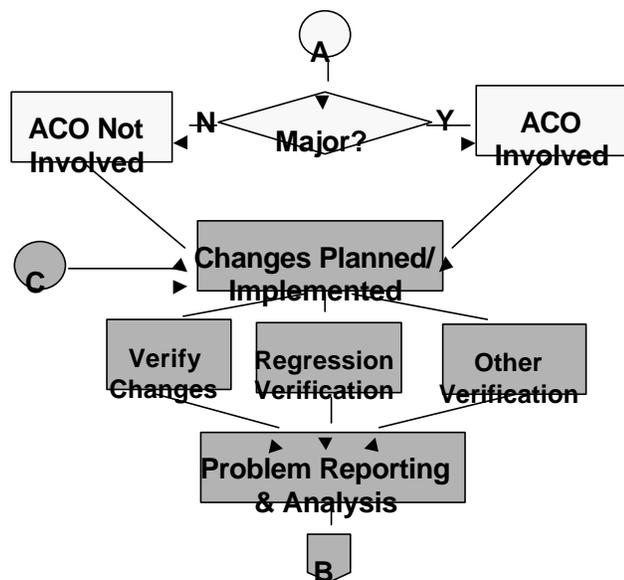
16

Big Picture Look at Software Changes (1/3)



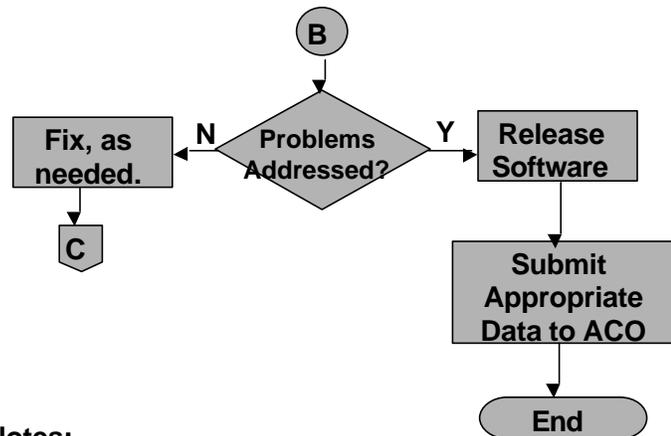
17

Big Picture Look at Software Changes (2/3)



18

Big Picture Look at Software Changes (3/3)



Notes:

- Reference Appendix C paper
- SQA, SCM continuous
- = “mini-development”

19

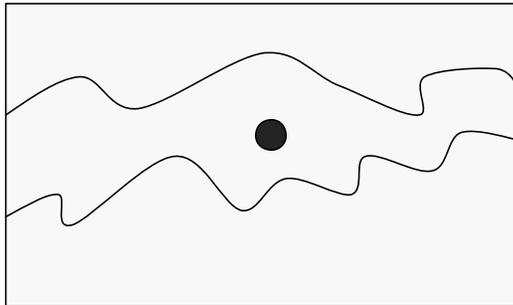
Purpose of Change Impact Analysis (1/2)

- Assess affects of the software change on system performance, safety, documentation, ...
- Assess the classification of the change (E.g., major, minor, significant, insignificant)
- Determine amount of rework and verification required
- Plan for the change (resources, cost, schedule, ...)

20

Purpose of Change Impact Analysis (2/2)

- Identify what is affected by the change



21



**Section 5 -
Discussion**

22

Overview of Section 5

- **Technical “meat” of the Notice**
 - “WHAT”, not “HOW”
- **3 Sub-Sections:**
 - 5a) Items to be addressed by CIA, as applicable
 - 5b) Examples of changes that could cause adverse affects
 - 5c) Updating data & verification

23

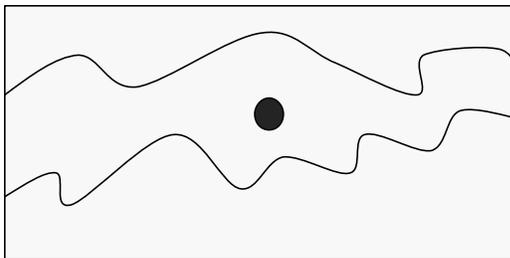
Section 5a: Potential Items to Be Addressed in CIA

- **Traceability Analysis**
- **Memory Margin Analysis**
- **Timing Margin Analysis**
- **Data Flow Analysis**
- **Control Flow Analysis**
- **Input/Output Analysis**
- **Development Environment & Process Analyses**
- **Operational Characteristics Analysis**
- **Certification Maintenance Requirements (CMR) Analysis**
- **Partitioning Analysis**

24

Traceability Analysis

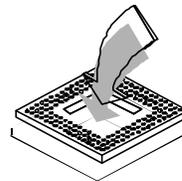
- **VERY IMPORTANT!**
- **Identifies areas affected by the software change:**
 - Requirements & Design Analysis
 - Code Analysis
 - Test Procedures and Cases Analysis



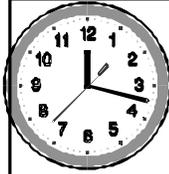
25

Memory Margin Analysis

- **Assure memory allocation requirements and margins are maintained.**
- **Examples of tasks:**
 - Estimate change to flash memory
 - Estimate change to RAM
 - Evaluate memory margins



26



Timing Margin Analysis

- **Assure timing margin issues are not introduced due to the change.**
- **Examples of tasks:**
 - Review timing requirements
 - Review CPU task scheduling requirements
 - Review interface timing requirements
 - Review changes to the timing margins (usually want at least 10% margin)
 - Review throughput change for each task

27

Data & Control Flow Analysis

- **DO-178B, Table A-7, Objective 8 requires data & control coupling for Levels A, B, and C software.**
- **Data & control flow analysis assesses changes in data & control flow and coupling between software components.**
 - Examples of software components are procedures and functions.
- **Data & control flow analysis also evaluates any adverse affects due to the change.**

28

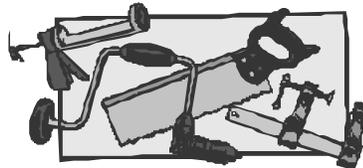
Input/Output Analysis

- I/O Analysis evaluates impact of the change on the interface with the external world:
- Examples of tasks:
 - Bus loading
 - External databus I/O
 - External hardwire I/O
 - Access to memory
 - Communication with hardware

29

Development Environment & Process Analysis

- Identifies changes in the environment or process that might have adverse affects on the system:
- Examples include changes to:
 - Compilers
 - Linkers
 - Loaders
 - Tools



30

Operational Characteristics Analysis

- Identifies adverse effects in the operational environment due to software changes.
- Examples of changes that could affect the operation of the product:
 - Gain changes
 - Limit changes
 - Filter changes
 - Interrupt changes
 - Exception handling changes
 - Fault mitigation changes

31

Certification Maintenance Requirements (CMR) Analysis

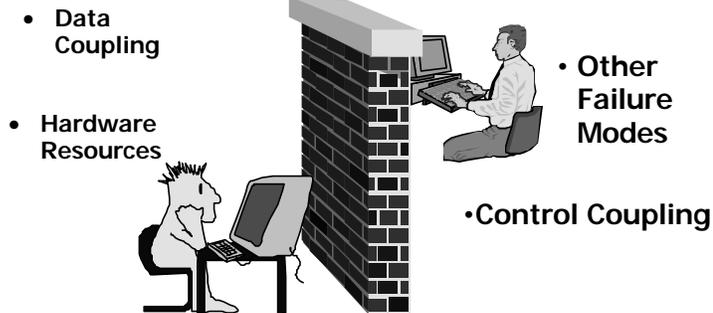
- Determines if the software change requires new or modified CMR.
- Example:
 - Assume the software change to the anti-skid systems increases the time that the brakes are applied during landing. This could result in more frequent maintenance of the brakes and tires.



32

Partitioning Analysis

- Determines the effect of the software change on the protective mechanisms.



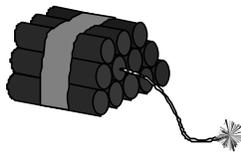
Section 5b. Examples of Adverse Affects (1/2)

- When performing the CIA activities, the focus is going to be on adverse affects. I.e., things that affect operation and safety.
- Section 5b provides examples of typical categories of change impact:
 - Change in safety-related information
 - Change in operational or procedural characteristics of the aircraft
 - New functions
 - Different interfaces
 - Significant change to life cycle data

34

Section 5b. Examples of Adverse Affects (2/2)

- Changes that have adverse impacts will likely lead to a “major” change classification.



35

5c. Updates and Verification

- Applicant updates necessary software life cycle data, whether the change is major or minor.
- Applicant verifies the software change to make sure there are no adverse effects. Example verification activities:
 - Reviews
 - Analyses
 - Regressions testing
 - Requirement-based testing
 - Flight testing

36

Changes to Requirements Most Common Change

- new
- modified
- deleted



37

Developer's Role For New/Changed Requirement (1/4)

- Perform CIA to assure that the new or changed requirement:
 - does not conflict with other requirements
 - is unambiguously stated and verifiable
 - is verified to meet requirements of software level
 - achieves desired functionality

38

Developer's Role For New/Changed Requirement (2/4)

- **Assure that the following are completed, as needed:**
 - **update the software architecture**
 - **change prologue headers**
 - **review changes against standards**
 - **update traceability (both forward and backward)**

39

Developer's Role For New/Changed Requirement (3/4)

- **Examine data elements to assure that new or changed code does not negatively impact existing functionality by:**
 - **examining all areas of the code that use the same variables as those in the changed or new code**
 - **re-examining variable declarations and interfaces**
 - **examining control flow to assure that the change does not negatively impact execution sequence or timing**

40

Developer's Role For New/Changed Requirement (4/4)

- **Assure that:**
 - Verification test cases for new and changed requirements exist
 - All requirements-based tests (normal and robust) that trace to new or changed requirements are run or re-run
 - Structural coverage is achieved for new or changed area, and still achieved for areas of code with dependencies
 - Verification record that documents the regression analysis exists

41

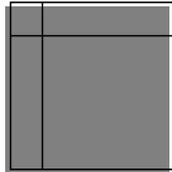
FAA's Roles For New/Changed Requirement



- **Oversee the applicant's activities, when the change is "major"**
- **Oversee designees**
- **Perform on-site or desk-top reviews, as needed**

42

Examples of Change Impact Analyses (1/3)



Tables/Forms



Automated
Tools

Change
Impact
Analysis

Fly-Today Company
Ground-Proximity Warning System
FTC-2100
Revision A
3/24/00

Formal Reports

43

Examples of Change Impact Analyses (2/3)

- **CIAs come in many forms**
 - Some Formal
 - Some Informal
- **No single correct format**
- **Extent of analysis depends on the change size and affected items**
- **Important to have the information available to make the necessary decisions**

44

Examples of Change Impact Analyses (3/3)

- See Appendix D
 - Modification to a TCAS unit
- See Appendix E
 - Modification to Lateral Guidance Software

45

Example of a Tool Used in CIA

The screenshot displays the JMS Process Manager interface with the following data tables:

ID	Description	File names	Revision	Status	Uplink Date
PIK_EFS_ICR_1423	DNPG - CONFIG MISMATCH CAS message change	e-5_360nfa1k1awc44rtgprv.c	3	APPROVED	18-Aug-1998 16:48:52
		e-5_360nfa1k1awc44rtgprv.c	14	APPROVED	18-Aug-1998 16:51:25
		e-5_360nfa1k1awc44rtgprv.c	7	APPROVED	18-Aug-1998 09:23:31
		e-5_360nfa1k1awc44rtgprv.c	25	TEST	18-Aug-1998 03:00:40
		e-5_360nfa1k1awc44rtgprv.c	116	TEST	18-Aug-1998 03:00:40
		e-5_360nfa1k1awc44rtgprv.c	87	TEST	18-Aug-1998 03:00:50

Specification	Description	Relationship
PIK_EFS_E4_310_MARNA1	E4S 110 Main Software Attached	Attached
PIK_EFS_E4_330D_PSR121214A1	E4S 330D Software Attached	Attached

ID	Description	Status
PIK_EFS_ICR_1423	DNPG - CONFIG MISMATCH CAS message change	UNDELETED
PIK_EFS_WP_28	Enhance Phase 45 (total)	CLOSED

Annotations on the left side of the screenshot:

- Code Files Effected (points to the 'File names' column)
- Rqmts Effected (points to the 'Revision' column)
- Change Request (CR) (points to the 'ID' column)
- Design Parts (code, req, or both) (points to the 'Design Parts' table)
- Test Change Request (TCR) (shows test effected) (points to the 'Status' column)
- Work Packet (WP) (used to group CRs into deliveries to customer) (points to the 'ID' column)

46

New Requirement Example

System Req'mts	Software Req'mts	Design	Code/ Module	Test
3.1.4	2.4.3	4.2.1	Correlate track	4
			Associate Track	4
	4.1.2	2.3.6	Track Update	4
	4.1.10a	2.3.10 2.3.9	DoScanFrame DeadReckonTrack	122
	4.1.10b	2.3.10 2.3.11	DoScanFrame Drop track	125

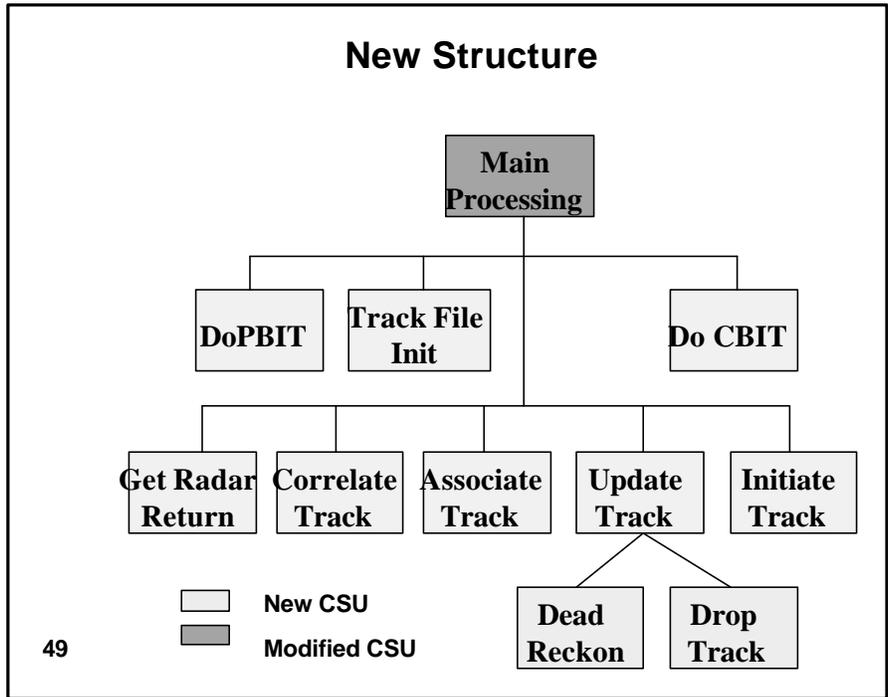
47

New Requirements

4.1.10a After each radar scan frame, all tracks that have not been updated by a radar return during that scan frame and have not been dead reckoned more than 5 consecutive scan frames should be dead-reckoned.

4.1.10b If a track has been dead reckoned for six consecutive scan frames, after the next scan frame the track should be dropped.

48



Program Changes

Procedure MainProcessing

```

DoPBIT
InitializeTrackFile
while radar operational
  while not EndScanFrame
    GetRadarReturn
    DoTrackCorrelation
    AssociateTrack
    if Associates True
      Update Track
    else
      InitiateTrack
    endif
  CheckEndFrame
endwhile
ScanFrame++
DoCBIT
endwhile
  
```

■ New or Modified Code

50

Procedure Update Track

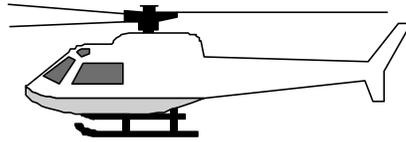
```
While TrackNum <= MaxTracks
  If Track(TrackNum). status ==Associate
    Track(TrackNum). Newx=radarx
    Track(TrackNum). Newy=radary
    Track(TrackNum). Newz=radarz
    Track(TrackNum).deadreckons =0
  else
    If EndScanFrame
      If Track(TrackNum).deadreckons <5
        Track(TrackNum). Newx=deadreckonx
        Track(TrackNum). Newy=deadreckony
        Track(TrackNum). Newz=deadreckonz
        Track(TrackNum).deadreckons ++
        If Track(TrackNum).deadreckons > 5
          DropTrack
        Endif
      endif
    Endif
  Endwhile
```

51

Exercise

- Given the new requirements for deadreckoning, please answer the following questions:
 - What data items need to be updated?
 - Using the data provided, was the change made correctly?
 - How would you determine which tests would have to be re-run?
 - What would data and control coupling analyses check for?

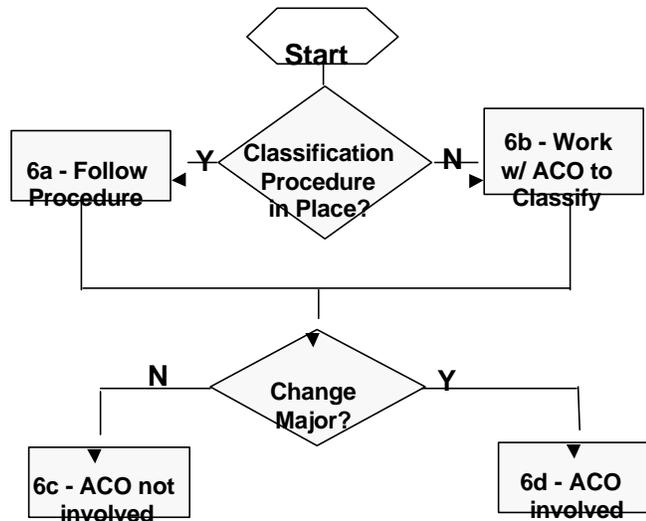
52



Section 6 - PROCEDURES

53

Overview of Section 6



54

6a - Applicant Has Classification Procedures (1/2)

- Procedures in place to classify changes as major or minor
- Reference FAR Part 21
- Procedures should be reviewed & approved by the ACO



55

6a - Applicant Has Classification Procedures (2/2)

- Procedures should contain a process for:
 - Using CIA to classify change
 - Reviewing/approving the classification
 - Addressing minor changes
 - Addressing major changes
 - Informing FAA (e.g., PSAC, SAS, report, ...)
 - Obtaining FAA concurrence on changes

56

Example Procedures

- **Appendix F**
 - AlliedSignal
- **Appendix G**
 - Honeywell (C130J) - DRAFT
- **Appendix H**
 - Draft process from Honeywell - Phoenix

57

6b - Without Classification Procedures

- **FAA more involved**
- **Applicant performs CIA (using Notice)**
- **Applicant proposes classification (major or minor) to FAA**
- **FAA reviews/accepts/modifies the classification**
- **Applicant & FAA follow 6c for minor changes and 6d for major changes**



58

6c - Minor Changes



- **Change performed without FAA involvement**
- **Data updated, as required**
- **Software Accomplishment Summary (SAS), Software Configuration Index (SCI), and/or other documents submitted to FAA on a periodic basis**

59

6d - Major Changes

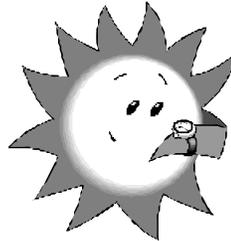


- **FAA and/or DER involved**
- **PSAC and/or CIA submitted to FAA as agreed upon**
- **SAS, SCI, and/or other agreed upon data submitted to ACO**
- **ACO and/or DER reviews and approves data, as needed**

60

Section 7 - Conclusion

- Notice is only a supplement AC 20-115B and DO-178B
- Guidelines only
- Variance from national policy should be coordinated with AIR-130



61

Summary

- Purpose of Notice is to provide guidelines for addressing changes to software.
- Intended to allow flexibility but encourage more standardization.
- Notice encourages use of CIA to serve as input into the major/minor classification.
- Send comments, questions, etc. to myself and/or Dennis.Wallace@faa.gov.

62

Notice on Software Change Impact Analysis

Appendix B

NOTICE

U.S. Department of Transportation
Federal Aviation Administration

N 8110.85

05/11/00

Cancellation

Date: 05/11/01

SUBJ: GUIDELINES FOR THE OVERSIGHT OF SOFTWARE CHANGE IMPACT ANALYSES
USED TO CLASSIFY SOFTWARE CHANGES AS MAJOR OR MINOR

1. PURPOSE. This notice provides guidelines to Aircraft Certification Office (ACO) engineers and Designated Engineering Representatives (DER) for overseeing an applicant's change impact analysis process. These guidelines are applicable to software changes related to type certificate (TC) approvals, amended type certificate (ATC) approvals, supplemental type certificate (STC) approvals, Parts Manufacturer Approvals (PMA), and Technical Standard Order (TSO) authorizations. This notice is for guidance purposes only and is supplemental to Advisory Circular 20-115B, "RTCA, Inc. Document RTCA/DO-178B," dated January 11, 1993.

2. DISTRIBUTION. This notice is distributed to the branch level in Washington Headquarters Aircraft Certification Service, section level in all Aircraft Certification Directorates, all National Resource Specialists (NRS), all Aircraft Certification Offices (ACO), all Manufacturing Inspection Offices (MIO), all Manufacturing Inspection District or Satellite Offices (MIDO/MISO), and all Flight Standards District Offices (FSDO). Additional limited distribution should be made to the Air Carrier District Offices, the Aeronautical Quality Assurance Field Offices, and the FAA Academy.

3. RELATED PUBLICATIONS.

a. Advisory Circular (AC) 20-115B, "RTCA, Inc. Document RTCA/DO-178B," dated January 11, 1993.

b. RTCA, Incorporated, document RTCA/DO-178B, "Software Considerations in Airborne Systems and Equipment Certification," dated December 1, 1992.

c. Title 14 Code of Federal Regulations (14 CFR), Part 21, "Certification Procedures for Products and Parts."

4. BACKGROUND.

a. On January 11, 1993, the FAA issued AC 20-115B which recognizes DO-178B as a means to secure FAA approval of digital computer software. DO-178B, section 12.1.1, identifies analysis activities to be performed for proposed software changes. DO-178B also implies that re-verification should be accomplished on all software changes and areas affected by those changes.

Distribution: A-W(IR)-3; A-X(CD)-4; A-FAC-0 (ALL),
A-FFS-7 (ALL); A-FFS-2,8 (LTD); AMA-220
(25 copies); AFS-600 (3 copies)

Initiated By: AIR-130

b. The purpose of this notice is to provide a standardized process to determine the impact of a software change on a system, in order to assure that safety is not adversely impacted. The notice focuses on the change impact analysis to determine the extent of certification authority involvement in the review of the changes and to determine the significance of the change in the overall project.

c. The change impact analysis may be used by an applicant to provide justification for the classification of a change as it relates to 14 CFR Parts 21.93 and 21.611. This notice does not contain examples of minor or major changes, but it does offer guidelines for analyzing the impact of software changes. Changes analyzed as minor (using the guidelines of this notice) for products previously approved under the TSO authorization process should be tested and verified by the applicant, but require no further oversight by the ACO engineer (per 14 CFR, Part 21). Likewise, changes analyzed as minor (using the guidelines of this notice) for products previously approved under the TC, STC, or ATC process should be tested and verified by the applicant and may be implemented for the software portions without further oversight by the ACO engineer or DER, if authorized, in accordance with 14 CFR, Part 21. However, the substantiation and description of the change(s) should still be submitted to the ACO in accordance with the delegation agreement.

5. DISCUSSION.

a. The applicant should identify the software changes to be incorporated in the product and perform a change impact analysis. The change impact analysis should follow a defined process to accomplish its purpose of determining the potential impact of the change on continued operational safety of the aircraft on which the product is installed. In the case of TSO authorized equipment, the analysis should identify the intended target aircraft environment which forms the basis for the analysis. This analysis also provides a basis for determining the extent of certification authority involvement. The following items should be addressed by the change impact analysis, as applicable:

(1) **Traceability analysis** to identify areas which could be affected by the software change. This includes the analysis of affected requirements, design, architecture, code, testing and analyses, as described below:

(a) *Requirements and design analysis* to identify software requirements, software architecture, and safety-related software requirements impacted by the change. Additionally, the analysis identifies any additional features and/or functions being implemented in the system, assures that added functions are appropriately verified, and assures that the added functions do not adversely impact existing functions.

(b) *Code analysis* to identify the software components and interfaces impacted by the change.

(c) *Test procedures and cases analysis* to identify specific test procedures and cases that will need to be re-executed to verify the changes, to identify and develop new or modified test procedures and cases (for added functionality or previously deficient testing), and to assure

that there are no adverse effects as a result of the changes. The absence of adverse effects may be verified by conducting regression testing at the appropriate hierarchical levels (e.g., aircraft flight tests, aircraft ground tests, laboratory system integration tests, simulator tests, bench tests, hardware/software integration tests, software integration tests, module tests), as appropriate for the software level(s) of the changed software.

(2) **Memory margin analysis** to assure that memory allocation requirements and acceptable margins are maintained.

(3) **Timing margin analysis** to assure that the timing requirements, central processing unit (CPU) task scheduling requirements, system resource contention characteristics, interface timing requirements, and acceptable timing margins are maintained.

(4) **Data flow analysis** to identify changes to data flow and coupling between components and assure that there are no adverse impacts.

(5) **Control flow analysis** to identify changes to the control flow and coupling of components and to assure that there are no adverse impacts.

(6) **Input/output analysis** to assure that the change(s) have not adversely impacted the input and output (including bus loading, memory access, and hardware input and output device interfaces) requirements of the product.

(7) **Development environment and process analyses** to identify any change(s) which may adversely impact the software product (e.g., compiler options or versions and optimization change; linker, assembler, and loader instructions or options change; or software tool change).

(8) **Operational characteristics analysis**, such as evaluation of changes to gains, filters, limits, data validation, interrupt and exception handling, and fault mitigation to assure that there are no adverse affects.

(9) **Certification maintenance requirements (CMR) analysis** to determine whether new or changed CMRs are necessitated by the software change.

(10) **Partitioning analysis** to assure that the changes do not impact any protective mechanisms incorporated in the design.

NOTE: The above list is not all inclusive and is dependent on the product for which the modification is being made.

b. The change impact analysis should determine whether the change could adversely affect safe operation of the system or product. The following are examples of areas that could have an adverse impact on safety or operation:

(1) **Safety-related information is changed.** For example:

(a) Previous hazards, as identified by the system safety assessment, are changed.

(b) Failure condition categories, as identified by the system safety assessment, are changed.

(c) Software levels are changed, particularly if the new software level is higher than the previous level.

(d) Safety-related requirements, as identified by the system safety assessment, are changed.

(e) Safety margins are reduced.

(2) **Operational or procedural characteristics of the aircraft are changed in a manner that could adversely affect flight safety as a result of the software change.** For example:

(a) Aircraft operational or airworthiness characteristics are changed.

(b) Flight crew procedures are changed.

(c) Pilot workload is increased.

(d) Situational awareness, warnings, and alerts are changed.

(e) Displayed information to make flight decisions is changed.

(f) Assembly and installation requirements are changed.

(g) Changes that affect equipment interchangeability and/or interoperability with other equipment.

(h) CMR's are changed or added.

(3) **New functions or features are added to the existing system functions that could adversely impact flight safety.**

(4) **Processors, interfaces, and other hardware components or the environment are changed in such a way that safety could be adversely affected.** Reference DO-178B, section 12.1.3.

(5) **Software life cycle data (e.g., requirements, code, architecture) is significantly changed in such a way that it could adversely affect safety.** For example:

(a) Software requirements, design, architecture, and code components (especially those affecting safety-related functions, partitioning, redundancy or safety monitors) are changed.

(b) Code (source, object, and executable object) components that perform a safety-related function or a component which provides an input to a component which performs a safety-related function are changed. (For purposes of this notice, a safety-related function is one which could potentially induce or allow a major, hazardous, or catastrophic failure condition to go undetected).

(c) Characteristics of the development environment impacting the executable object code are changed.

(d) Memory allocation requirements are changed in such a way that memory margins are adversely impacted (e.g., less than 5% margin remaining).

(e) Timing requirements are changed in such a way that timing margins are adversely impacted (e.g., margins are unpredictable or less than 10% margin remains).

(f) Input/output requirements (e.g., bus loading) are changed in such a way that input or output performance is adversely impacted (e.g., less than 5% margin remains).

(g) Data and control coupling characteristics are adversely impacted (e.g., to the extent that more than 50% of the coverage analysis must be redone).

(h) Interface characteristics are changed.

c. Additionally, the following items should be identified in the change impact analysis:

(1) **Updates** that will be needed to assure that the software change(s) is incorporated in the appropriate software life cycle data, including requirements, design, architecture, source and object code, and traceability.

(2) **Verification activities** that will be needed to verify the changes and to verify that there are no adverse effects on the system. The change impact analysis should address how changes which could adversely affect safe operation of the system or aircraft will be verified, such that the changed and unchanged software will continue to satisfy their requirements for safe operation. These verification activities may include reviews, analyses, regression testing, requirements-based testing, flight testing, etc., including re-evaluation of existing analyses, re-execution of existing tests, and new test procedures and cases (for added functionality or previously deficient testing).

6. PROCEDURES. Each project involving software changes has different needs. This section outlines procedures for the ACO engineer or DER, if authorized, to consider with the applicant when addressing software changes.

a. The applicant may define and follow a procedure for classifying software changes as major or minor and should seek ACO review, feedback, and approval for that procedure. As a minimum, any such procedure should address the following before being implemented:

(1) The applicant's process for using the change impact analysis (as addressed in section 5 of this notice) to justify a minor or major change classification and the criteria used by the applicant to make the change classification.

NOTE 1: The extensiveness and formality of the change impact analysis will vary by complexity, criticality, and extensiveness of the change. The change impact analysis may be in-depth for complex, highly critical systems but may be briefer and less rigorous for less complex or less safety critical systems or less extensive changes.

NOTE 2: The applicant's documentation should address the categorization of the change as minor or major, per the appropriate regulations, (e.g., Part 21.93 and/or Part 21.611) in order to obtain FAA agreement on the change classification.

(2) The applicant's process to review and approve the change classification (e.g., DER review and approval).

(3) The process to be followed for a minor change determination (reference section 6c of this notice).

(4) The process to be followed for a major change determination (reference section 6d of this notice).

(5) The process for informing the FAA of all proposed software changes and their proposed classifications.

(6) The process for obtaining FAA concurrence with the proposed classifications.

NOTE: Once ACO approval of the software change classification procedure has been granted, the applicant should follow the procedure for all proposed software changes. Deviations from the approved procedure should obtain FAA concurrence.

b. If the applicant does not have an FAA approved software change classification procedure, the applicant should inform the FAA and/or DER, if applicable, that a software change is being planned. In these cases, the applicant should perform the following activities:

(1) Perform a change impact analysis, using the guidelines in section 5 of this notice.

(2) Propose a major or minor classification for the change (based on the change impact analysis and safety implications as stated in section 5 of this notice) and seek FAA feedback and concurrence on the classification.

(3) Support any proposed minor classification with rationale about the absence of safety impact and/or the limited scope of the change, and the proposed method of verifying the change. After the FAA has agreed to the applicant's data and rationale, the applicant may proceed without further FAA oversight for minor changes (reference section 6c of this notice).

(4) Submit the appropriate documentation to the FAA for major changes (reference section 6d of this notice).

c. For minor changes, the ACO oversight of the development process should involve approval and periodic review of the applicant's change impact analysis process and associated criteria for making a major/minor determination with respect to the relevant regulations. Once the change strategy and the change itself have been performed, the strategy should be documented in the Software Accomplishment Summary (SAS). New, modified, and re-used software life cycle data should also be identified in the Software Configuration Index (SCI). For minor changes, submittals of the SAS and SCI to the ACO should be per agreement with the ACO.

NOTE 1: When applicable, DERs should be involved in the change classification procedure and oversight of the company's adherence to that procedure.

NOTE 2: Equipment containing changes that are classified by the manufacturer as minor but not yet concurred with by the ACO or DER, when authorized, should be withheld from installation on flight aircraft until the ACO concurs with the classification.

d. For major changes, the ACO engineer and/or DER, if authorized, should review the applicant's Plan for Software Aspects of Certification or other summary of change impact analysis data and the applicant's proposed strategy for addressing the change issues. Once the change strategy and the change itself have been carried out, the ACO engineer and/or DER, if authorized, should assure that the strategy is documented and submitted in the SAS. New, modified, and re-used software life cycle data should also be identified in the SCI and submitted to the ACO engineer and/or DER, if authorized to approve major changes.

NOTE: In many cases, a change process may already be in place to address major, minor, significant, insignificant, small, simple, etc. changes. The applicant's change impact analysis activities (in accordance with this notice) should fit within their already existing framework in order to avoid unnecessary or inappropriate activities.

7. CONCLUSION. The information and procedures described in this notice promote clarification and consistent application of AC 20-115B for the approval of changes to the software or its environment in the airborne system and equipment. This notice does not replace or supersede AC 20-115B or DO-178B.

David Hempe
Acting Manager, Aircraft Engineering Division,
Aircraft Certification Service

Draft Paper – Planning for Software Changes in Safety-Critical Systems

Appendix C

PLANNING FOR SOFTWARE CHANGES IN SAFETY-CRITICAL SYSTEMS

DRAFT DOCUMENT – March 14, 2000

Leanna K. Rierson, Federal Aviation Administration, Washington, D.C.

Abstract

This paper presents activities that should be considered when planning for software changes to safety-critical systems. The paper focuses on the aviation industry but could also apply to other safety-critical fields, such as medical or nuclear. This is a ***draft paper*** which will be modified and polished for the Digital Avionics Systems Conference to be held in October 2000. If you have comments on the paper or desire to obtain a finalized version, send an e-mail to Leanna.Rierson@faa.gov.

Introduction

A majority of software projects are changes to already existing software, rather than development of totally new software. Software changes occur to add new functions, to correct problematic areas, or to change existing functions. When software changes occur in safety-critical systems, great care must be taken. The changes must be carefully planned, assessed, controlled, and tested. This paper will address the typical steps involved in a software change process. The change process should be planned up-front and carefully followed when the actual changes occur.

Planning for Change

Changes to software are a way of life in today's environment. A 1988 paper presented in *Proceedings of the Sixth Annual Pacific Northwest Software Quality Conference* emphasized that 40 to 70% of the total life-cycle costs associated with a software intensive

system are maintenance activities; i.e., changes made to software [1].

The Software Engineering Institute (SEI) "Workshop on the State of the Practice in Dependably Upgrading Critical Systems" explore ways to dependably upgrade software. One area of focus at the workshop was the "design for upgrade" concept [2].

In the original software development, developers and testers should plan for future changes. Putting a change process in place, developing automatic regression tests, and developing a test library can help address future changes. The SEI workshop emphasized that "most 'new' systems are updates to existing (legacy) systems rather than completely new systems" [2]. Therefore, designing for upgrade can really help address future changes. The following items should be considered when designing for upgrade [2]:

- Architectures that include ease of changes should be considered.
- Automated tools for design and verification should be implemented.
- Methods to identify and isolate the impact of an upgrade should be explored.
- Tools to identify the dependencies among different system components should be used.
- Cost models should be defined to anticipate changes and their impact.

Planning and documenting a change process is critical to addressing software changes. The following figure shows the eight steps that are typically included in a change

process. Each step should be carefully planned during the original development and modified to meet the program’s specific needs, as appropriate. The change process should be documented. When an actual change occurs the plan will be followed and the specific tests, resources, etc. needed to implement the change can also be more specifically planned, based on the change impact analysis. Each of the eight steps of the change process will be described in this paper. Emphasis will be placed on step 4 – the testing process.

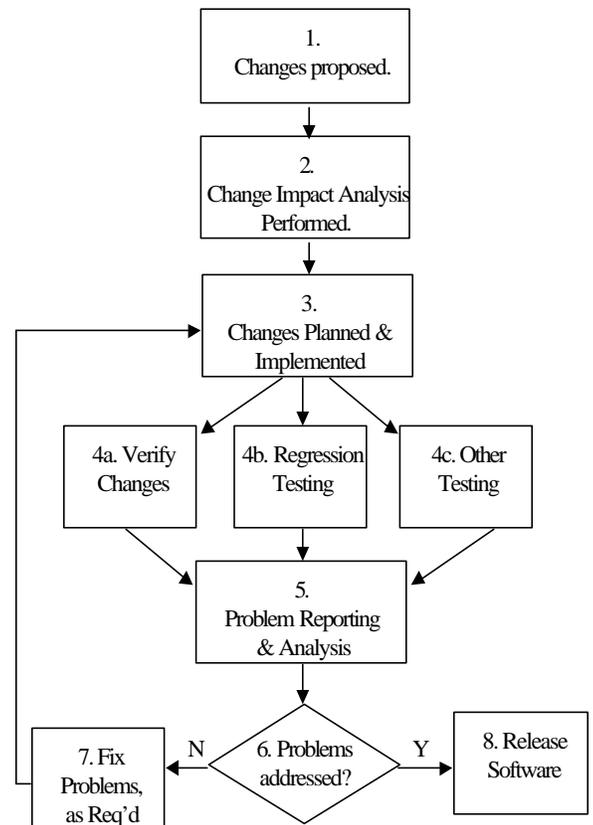
Step 1: Changes are Proposed.

The first step of the change process is to receive the proposed changes. These changes should be documented in a consistent format. The change requests may include requests to fix bugs, to add new features, to modify existing features, etc.

Step 2: Perform a Change Impact Analysis.

It is typically not technically possible or financially feasible to address all of the proposed changes. In order to determine the impact of the change, both technically and economically, a change impact analysis should be performed. In a change impact analysis each change is systematically analyzed to determine the potential impact. Requirements traceability will help to determine the impact in the overall software project [3].

The Federal Aviation Administration has recently completed policy on change impact analysis in a document entitled, “GUIDELINES FOR THE OVERSIGHT OF SOFTWARE CHANGE IMPACT ANALYSES USED TO CLASSIFY SOFTWARE CHANGES AS MAJOR OR MINOR.” The purpose of the FAA paper was to help developers of safety-critical software and the FAA to assess the extent of the change. This extent then drives the amount of regression testing, reviews, inspections, etc.



DRAFT

The FAA determined that the following items should be addressed by the change impact analysis, as applicable [4]:

(1) **Traceability analysis** to identify areas which could be affected by the software change. This includes the analysis of affected requirements, design, architecture, code, testing and analyses, as described below:

Requirements and design analysis to identify software requirements, software architecture, and safety-related software requirements impacted by the change. Additionally, the analysis identifies any additional features and/or functions being implemented in the system, assures that added functions are appropriately verified, and assures that the added functions do not adversely impact existing functions.

Code analysis to identify the software components and interfaces impacted by the change.

Test procedures and cases analysis to identify specific test procedures and cases that will need to be re-executed to verify the changes, to identify and develop new or modified test procedures and cases (for added functionality or previously deficient testing), and to assure that there are no adverse effects as a result of the changes. The absence of adverse effects may be verified by conducting regression testing at the appropriate hierarchical levels (e.g., aircraft flight tests, aircraft ground tests, laboratory system integration tests, simulator tests, bench tests, hardware/software integration tests, software integration tests, module tests), as appropriate for the software level(s) of the changed software.

(2) **Memory margin analysis** to assure that memory allocation requirements and acceptable margins are maintained.

(3) **Timing margin analysis** to assure that the timing requirements, central processing unit (CPU) task scheduling requirements, system resource contention characteristics,

interface timing requirements, and acceptable timing margins are maintained.

(4) **Data flow analysis** to identify changes to data flow and coupling between components and assure that there are no adverse impacts.

(5) **Control flow analysis** to identify changes to the control flow and coupling of components and to assure that there are no adverse impacts.

(6) **Input/output analysis** to assure that the change(s) have not adversely impacted the input and output (including bus loading, memory access, and hardware input and output device interfaces) requirements of the product.

(7) **Development environment and process analyses** to identify any change(s) which may adversely impact the software product (e.g., compiler options or versions and optimization change; linker, assembler, and loader instructions or options change; or software tool change).

(8) **Operational characteristics analysis**, such as evaluation of changes to gains, filters, limits, data validation, interrupt and exception handling, and fault mitigation to assure that there are no adverse affects.

(9) **Certification maintenance requirements (CMR) analysis** to determine whether new or changed CMRs are necessitated by the software change.

(10) **Partitioning analysis** to assure that the changes do not impact any protective mechanisms incorporated in the design.

Note: The above list is not all inclusive and is dependent on the product for which the modification is being made.

The FAA also determined that the change impact analysis should determine

DRAFT

whether the change could adversely affect safe operation of the system or product. The following are examples of areas that could have an adverse impact on safety or operation [4]:

(1) Safety-related information is changed. For example:

- Previous hazards, as identified by the system safety assessment, are changed.
- Failure condition categories, as identified by the system safety assessment, are changed.
- Software levels are changed, particularly if the new software level is higher than the previous level.
- Safety-related requirements, as identified by the system safety assessment, are changed.
- Safety margins are reduced.

(2) Operational or procedural characteristics of the aircraft are changed in a manner that could adversely affect flight safety as a result of the software change. For example:

- Aircraft operational or airworthiness characteristics are changed.
- Flight crew procedures are changed.
- Pilot workload is increased.
- Situational awareness, warnings, and alerts are changed.
- Displayed information to make flight decisions is changed.
- Assembly and installation requirements are changed.
- Changes that affect equipment interchangeability and/or interoperability with other equipment.

(3) New functions or features are added to the existing system functions that could adversely impact flight safety.

(4) Processors, interfaces, and other hardware components or the environment are changed in such a way that safety could be adversely affected.

(5) Software life cycle data (e.g., requirements, code, architecture) is significantly changed in such a way that it could adversely affect safety. For example:

- Software requirements, design, architecture, and code components (especially those affecting safety-related functions, partitioning, redundancy or safety monitors) are changed.
- Code (source, object, and executable object) components that perform a safety-related function or a component which provides an input to a component which performs a safety-related function are changed. (For purposes of this notice, a safety-related function is one which could potentially induce or allow a major, hazardous, or catastrophic failure condition to go undetected).
- Characteristics of the development environment impacting the executable object code are changed.
- Memory allocation requirements are changed in such a way that memory margins are adversely impacted (e.g., less than 5% margin remaining).
- Timing requirements are changed in such a way that timing margins are adversely impacted (e.g., margins are unpredictable or less than 10% margin remains).
- Input/output requirements (e.g., bus loading) are changed in such a way that input or output performance is adversely impacted (e.g., less than 5% margin remains).
- Data and control coupling characteristics are adversely impacted (e.g., to the extent that more than 50% of the coverage analysis must be redone).
- Interface characteristics are changed.

Lastly, the FAA determined that the following items should be identified in the change impact analysis [4]:

(1) **Updates** that will be needed to assure that the software change(s) is incorporated in the appropriate software life cycle data, including requirements, design,

DRAFT

architecture, source and object code, and traceability.

(2) **Verification activities** that will be needed to verify the changes and to verify that there are no adverse effects on the system. The change impact analysis should address how changes which could adversely affect safe operation of the system or aircraft will be verified, such that the changed and unchanged software will continue to satisfy their requirements for safe operation. These verification activities may include reviews, analyses, regression testing, requirements-based testing, flight testing, etc., including re-evaluation of existing analyses, re-execution of existing tests, and new test procedures and cases (for added functionality or previously deficient testing).

Note: A checklist and planning worksheet to assist in change impact analysis can be found on the world-wide web at www.processimpact.com/process-assets/impact_analysis.doc [3].

Step 3: Plan and Implement Changes

Once a change impact analysis has been performed, the development team should plan and implement the changes. The planning should include scheduling reviews and tests, as well as changing the requirements, design, code, etc.

A software modification can be handled like a “mini-development”. Plans would need to be put in place for development, quality assurance, configuration management, and testing. Once the plans are in place, the changes may be implemented.

Step 4: Testing the Change

Once the changes have been implemented, they must be verified and validated. The flow-chart in this paper breaks

the “testing” process into three tasks (4a, 4b, and 4c). Each will be described below:

Task 4a: Verify Changes

This task includes what is commonly referred to as the “verification” activities (i.e., reviews, analyses, inspections and walkthroughs). Edward Kit describes verification as a “human examination or review of the work product” [5]. Basically, verification is the process of checking requirements, design, code, test cases, etc. for accuracy.

Task 4b: Perform Regression Testing

Webster’s dictionary defines regression as “a trend or shift toward a lower or less perfect state.” Boris Beizer defines regression testing as “any repetition of tests (usually after software or data change) intended to show that the software’s behavior is unchanged except insofar as required by the change to the software or data [6]. Software progresses through several versions before one is ready for release. Regression testing is performed on each version of software. The biggest issue is how to determine which software tests in Version N need to be run, if they were already tested in Version N-1. Any specific change can (a) fix only the problem that was reported, (b) fail to fix the problem, (c) fix the problem but break something that was previously working, or (d) fail to fix the problem and break something else [7]. It is typically not possible to re-run every test on every version of software, so care must be used in determining which tests should be run on the interim versions.

Joy Shafer’s paper entitled, “Regression Testing Basics” provides some insight that is useful for planning regression testing. She writes that “regression testing finds many bugs. Studies have shown that changes and error corrections tend to be much more error prone

DRAFT

than the original code in the program, in much the same way that most copy errors that make it to print were introduced during edits rather than in the original draft” [8]. Shafer goes on to list some of the most common types of regression test [8]:

- Bug verification tests – run to verify that the fix for a bug addresses the problem and doesn’t introduce additional problems.
- Build acceptance tests – tests run to make sure that a build is ready to go to the test team.
- Regression test pass with a regression test suite – running regressions tests that have been automated.
- Regression test pass on closed bugs – rerunning the regression tests even after the bugs have been “fixed”.
- Regression test pass without a test suit – manually running regression tests.

Shafer also recommends that the most important tests be run first in order to quickly validate operation and assess risks. She also encourages the use of test suites to help reduce the time factor in re-running regression tests [8].

Another paper on regression testing by Christian Molnar discussed what should be tested and when it should be tested. Molnar recommends that regression testing be run in parallel with other development activities. This approach helps to address errors early. However, this approach also poses the question of “how many tests need to be re-run?” That is, as the project matures, do all regressions tests need to be re-run? Mohlar offers these “general rules” for applying regression testing [9]:

- A test that has passed twice should be considered as regressed, unless turmoil exists in the code.
- A test that has failed once should not be re-executed unless the developer informs the test team that the defect has been fixed.
- For tests that have already passed once, the second execution should be reserved for the

final regression pass, unless turmoil in the code indicates otherwise.

- The final regression test should not consist of 30 to 40% of the total number of tests in the suite.

For safety-critical systems, I tend to disagree with Mohlar’s 4th rule. For safety-critical systems, all regressions tests should be run on the final version prior to release.

Task 4c: Perform Other Testing

In addition to the regression testing, there may be other types of tests to be run on the software and system. For example, requirements-based tests, acceptance tests, flight tests, structural coverage analysis, etc. may need to be performed. These additional tests may vary from project to project, depending on the extent of the change and the function(s) affected. These tests should be planned after the change impact analysis is performed and agreed upon early in the project.

Task 5: Problem Reporting and Analysis

Any problems detected during Task 4 should be documented in a problem report. In his paper entitled, “The Bug Reporting Process,” Matt Baskett emphasized the importance of quality problem reports. He stated that the report should include a title, severity, description, steps to reproduce a bug, actual results of the test, expected results of the test [10]. Additionally, the problem report should include a place to identify affected documents, problem analyses, and proposed resolution.

Task 6: To Determine if Necessary Problems Are Addressed

At some point in the program, it should be assessed if all of the necessary problems have been addressed. Most projects finish with open problem reports; however, there needs to

DRAFT

be some kind of method to determine which problem must be addressed and which ones are okay to leave open. In safety-critical systems, the software engineer, the systems engineer, and a safety expert should assess each open problem report to determine if it affects the safety of the system. Safety-related problems should be addressed prior to release of the product.

Task 7: Fix the Problems, As Required

As mentioned above, not all problems are fixed; however, when problems are fixed, they will need to again go through the change and testing process.

Task 8: Release the Software

Once the necessary changes have been implemented, problems have been addressed, and tests have been run, the product is ready for release.

Configuration Management and Quality Assurance

Throughout the change process, configuration management and quality assurance should be applied. It is important to know exactly what version of software is being tested and to have independent reviewers overseeing the process.

Summary

This paper shows the planning that must occur for software changes to safety-critical products. Software changes are a way of life – good planning is essential to properly addressing changes. The testing process is a particularly important part of the change process and must be carefully planned. In most projects, a general change process is planned during the original development. The more

specific details (i.e., resources, types of tests, etc.) are planned once a change impact analysis has been completed.

References

1. T.J. Ostrand and E.J. Weyuker, "Using Data Flow Analysis for Regression Testing," 233-247. *Proceedings of the Sixth Pacific Northwest Software Quality Conference, 1988.*
2. David Gluch and Charles Weinstock (editors), "Workshop on the State of the Practice in Dependably Upgrading Critical Systems," SEI, 1997. Report #: CMU/SEI-97-SR-014 (web-site: <http://www.sei.cmu.edu/pub/documents/97.reports/pdf/97sr014.pdf>).
3. Karl Wiegers, "Karl Wiegers Describes 10 Requirements Traps to Avoid," *STQE Magazine*, Jan/Feb 2000 (web-site: <http://stqemagazine.com/>).
4. Leanna Rierson (leader), "GUIDELINES FOR THE OVERSIGHT OF SOFTWARE CHANGE IMPACT ANALYSES USED TO CLASSIFY SOFTWARE CHANGES AS MAJOR OR MINOR," Federal Aviation Administration policy, December 1999.
5. Edward Kit. Software Testing in the Real World, Addison-Wesley, 1995.
6. Boris Beizer, Software Testing Techniques (2nd edition), Thomas Computer Press, 1990.
7. Whittaker, James A., "What is Software Testing? And Why is it So Hard?" Software, IEEE, January/February, 2000.
8. Joy Shafer, "Regression Testing Basics," Data Dimensions, May 1998 (web-site: <http://www.stlabs.com/testersnet/docs/regression.htm>).
9. Christian Molnar, "Regression Testing: 'What' to Test and 'When'", Data Dimensions (web-site: <http://www.stlabs.com/testersnet/docs/regress9.htm>).
10. Matt Baskett, "The Bug Reporting Process," Data Dimensions (web-site: <http://www.stlabs.com/testersnet/docs/bugreport.htm>).

Example Change Impact Analysis – AlliedSignal TCAS

Appendix D

Request for Minor Software Modification

Requestor: Barry Webb
Date: 05/06/99
Unit Model: TPU 66A
TSO: C118
Unit P/N: 066-01145-0101
Current SW P/N: 206-00291-0111
New SW P/N: 206-00291-0112
SW MOD: 01/12
SW CID: 716-00120-0112

Complete Description of Change: The TPU 66A TCAS processor has an interface anomaly when connected to a competitor's newly released digital radar altimeter. The current TCAS software expects the altitude to trip to 4000 feet and provide SSM=NCD when the radar altimeter is out of range from the ground. The newly developed altimeter provides the actual value received, regardless of strength, and sets the SSM field to NCD when the return signal is too low. In this case the TCAS system will fail the altimeter. The failure only occurs at radar altitudes above 2500 but less than 4000 feet. The proposed resolution to the anomaly is for the TCAS software to be modified to consider the radar altimeter out of range from the ground whenever the SSM is set to NCD. This modification is already implemented in AlliedSignal's TCAS II system (TPU 67A processor with SW MOD 01/12).

Additionally, a modification is desired in testing the video board in the unit. A nuisance BITE failure can occur due to an inopportune task swap between two instructions during the board test (clearing a hardware bit and entering a MultiTask! critical region). The modification desired is to swap the two instructions to prevent this nuisance failure.

Responsible Engineer: _____

SQA Acceptance of Request: _____

Date: _____

Concurrence of Minor Software Modification

CO Number:

Responsible Engineer: _____

SQA Concurrence: _____

Date: _____

Justification of Minor Software Modification

Effect of Change on:	Analysis
Traceability	One requirement (para. 3.3.2.1.1.3.7.5 1d) and associated test (bbet_004) will have to be modified, but traceability will be unaffected.
Memory Margin	No impact
Timing Margin	No impact
Data Flow	No impact
Control Flow	No impact
Input / Output (bus loading)	No impact
Requirements and Design	Defect report #572 opened against the System Requirements Document removing the requirement to monitor radio altitude with SSM
Code	Module - i4hztask.c, function - rad_alt_update, delete 4 lines (remove check for > 4000 ft. when NCD and mark altitude as not credible and not invalid) Module - vbrdtst.c, function - video_board_md3x_bite, swap the order of two instructions
Development environment and process	No impact
Documentation	System requirement listed above under traceability
Operational characteristics	No impact
Identify features and functions being added	No impact
Identify test procedures	bbet_004 to validate radio altimeter modifications and MOPS high density tests 24321, 24332 and 2434 as regression tests for the system
Other (describe)	None

Change Impact Analysis	Yes	No
Previous hazards as identified by the system safety assessment are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Failure condition categories as identified by the system safety assessment are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Software levels are changed, especially if new level is A, B or C.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Safety related requirements as identified by the system safety assessment are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Software requirements, architecture and code components, especially those affecting safety related functions, partitioning, redundancy or safety monitors are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Code (source, object, executable) components that perform a critical function or a component which provides an input to a component which performs a critical function are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Characteristics of the development environment impacting the executable object code are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Memory allocation requirements are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Memory margin is adversely impacted.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Timing requirements are changed	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Timing margin is adversely impacted.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Bus loading (I/O) requirements are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Bus loading margin is adversely impacted.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Data and control coupling characteristics are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Protection mechanisms to ensure partitioning are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Processors or programmable devices are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Aircraft operational characteristics are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Flight crew procedures are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Flight Manual Supplement revision is required	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Aircraft worthiness characteristics are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Safety margins are reduced.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Pilot workload is increased.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Situational awareness, warnings and alerts are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Displayed information to make flight decisions is changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Assembly and installation requirements are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Example Change Impact Analysis – Honeywell Lateral Guidance Software

Appendix E

Software Change Impact Analysis for Lateral Guidance Software Error Correction

1. Identify the aircraft on which the product is installed. [For TSO articles, identify the intended target aircraft environment].	Intended target aircraft is the Alenia C-27J.
2. Describe why the software change is needed.	Correct software error related to flight plan leg transitions. The error can cause waypoint transition data to be incorrectly calculated for all legs in a flight plan and can result in an incorrect turn radius calculation for flight plan leg transitions.
3. Describe the software change to be implemented.	Add one line of code to initialize a variable in lateral guidance.
4. Requirements/design/code traceability analysis Note 1: Items that are affected include items that are changed, added, or deleted. Note 2: Safety related requirements/design/code are those related to monitoring, redundancy, protection/partitioning, etc.	
4.1. Identify the affected requirements in the SRD and SDD	No requirements are affected by this change. This software change will bring the system into compliance with current requirements.
4.2. Identify the affected architectural design	There are no architectural design affects due to this change.
4.3. Identify the software components (procedures/functions) to be changed	Add an initialization statement to: Ada package LG_DEF_TRANS, procedure DEFINE_LATERAL_TRANS_DATA
4.4. Identify any affects to safety related requirements	None
4.5. Identify any affects to safety related architectural design	None
4.6. Identify any affects to safety related procedures/functions	None
4.7. Identify any new features or functions to be implemented	None
4.8. Identify any affects to hardware design	None

<p>5. Verification traceability analysis Note 1: Verification activities can include review, analysis, regression testing, re-execution of existing tests, requirements-based testing, structural testing, etc.</p>	
<p>5.1. Identify the affected (i.e., changed, added, or deleted) verification procedures including any new procedures necessary for new features/functions or previously deficient verification</p>	<p>There is no affect to existing verification procedures. A new verification procedure will be developed to ensure proper implementation of the proposed software change.</p>
<p>5.2. Identify the verification activities which will be performed to ensure that:</p> <ul style="list-style-type: none"> • the changes are verified • the changed software complies with product requirements • there are no adverse effects as a result of the change • new features or functions are verified • new features or functions do not adversely impact existing features or functions 	<p>Upon completion of the proposed software change, selected existing verification procedures will be run to ensure that the system has not been adversely affected by the change. The verification procedures that will be selected for this task will be those that are designed to verify the portion of the system affected by the change. Additionally, a new verification procedure will be developed to ensure proper implementation of the proposed software change. Following successful completion of these tests targeted to the area of change, the software will be tested against the full suite of verification procedures.</p>
<p>6. Memory margin analysis</p>	
<p>6.1. Estimate the change to the flash memory</p>	<p>A negligible amount (<0.1%) of additional flash memory will be used.</p>
<p>6.2. Estimate the change to the RAM memory</p>	<p>None</p>
<p>6.3. Determine if acceptable memory allocation margins are maintained (e.g., memory margin would be adversely affected if there is less than 5% margin remaining)</p>	<p>No adverse impact. Approximately 87% of flash memory margin will remain.</p>
<p>7. Timing margin analysis</p>	
<p>7.1. Estimate the change to throughput margin for each affected task</p>	<p>A negligible amount (~10 micro-seconds per initialization) of additional throughput will be used.</p>
<p>7.2. Determine if acceptable timing allocation margins are maintained (e.g., timing margin would be adversely affected if it is unpredictable or there is less than 10% margin remaining)</p>	<p>No adverse impact. Approximately 98% of throughput margin will remain in the affected (1sec.) task.</p>
<p>8. Data coupling analysis</p>	
<p>8.1. Analyze data coupling changes to identify software components (procedures/functions) which would be affected by the proposed software change</p>	<p>The change to be made does not have data coupling issues, as it is simply an initialization of the variable to force determination of its proper value.</p>
<p>8.2. Determine if data coupling changes have resulted in any adverse impacts (e.g., to the extent that more than 50% of the coverage analysis must be redone)</p>	<p>No adverse impact.</p>

9. Control coupling analysis	
9.1. Analyze control coupling changes to identify software components (procedures/functions) which would be affected by the proposed software change	The change to be made does not have control coupling issues, as it is simply an initialization of the variable to force determination of its proper value.
9.2. Determine if control coupling changes have resulted in any adverse impacts (e.g., to the extent that more than 50% of the coverage analysis must be redone)	No adverse impact.
10. External input/output interface analysis	
10.1. Identify changes to the external interfaces (e.g., databus I/O, hardwired I/O, etc.)	None.
10.2. Determine if I/O changes have resulted in any adverse impacts (e.g., I/O margin would be adversely affected if there is less than 5% margin remaining)	No adverse impact.
11. Hardware component analysis	
11.1. Identify processor or interface hardware changes	None.
11.2. Determine if hardware component changes have resulted in any adverse safety impacts	No adverse impact.
12. Development environment analysis	
12.1. Identify changes to development tools	None
12.2. Identify changes to the development environment (e.g., build process including compiler versions, optimization, options; linker, assembler, loader instructions)	None
13. Software operational characteristics analysis	
13.1. Identify changes to gains, filters, limits, data validation, interrupt and exception handling, and fault mitigation	None
13.2. Determine if software operational characteristic changes have resulted in any adverse impacts	No adverse impact.
14. Safety analysis	
14.1. Identify affected failure conditions, severity classifications, and probability of occurrence	Implementation of this change will correct the error described in item 2 above. This change will have no affect on other failure conditions, severity classifications, or probabilities of occurrence.
14.2. Identify affected software criticality levels	This change will have no affect on the software criticality level.

15. Aircraft operational analysis	
15.1. Identify affects on the aircraft operational requirements or limitations	The proposed software change will correct an existing problem related to improper flight plan leg transitions. Since this is a correction to an existing problem, there will be a positive affect on aircraft operation, airworthiness, and crew workload. Additionally, there will be no affect on flight crew procedures, situational awareness, flight crew advisories, installation requirements, interchangeability, interoperability, or CMR's.
15.2. Identify affects on airworthiness characteristics	
15.3. Identify affects on flight crew procedures and workload	
15.4. Identify affects on situational awareness including display of cockpit information	
15.5. Identify affects on flight crew advisories, cautions, and warnings	
15.6. Identify affects on installation requirements or limitations	
15.7. Identify affects on equipment interchangeability and/or interoperability with other equipment	
15.8. Identify affects on certification maintenance requirements (CMR)	
16. Life cycle data analysis	
16.1. Plan for Software Aspects of Certification	PSAC will be updated to describe this software change.
16.2. Software Development Plan	No affect
16.3. Software Verification Plan	No affect
16.4. Software Configuration Management Plan	No affect
16.5. Software Quality Assurance Plan	No affect
16.6. Software Requirements Standards	No affect
16.7. Software Design Standards	No affect
16.8. Software Code Standards	No affect
16.9. Software Requirements Data	No affect
16.10. Software Design Description	No affect
16.11. Source Code	Add one line of code to initialize a variable in lateral guidance.
16.12. Executable Object Code	Will change based on the source code change.
16.13. Requirements traceability	No affect
16.14. Software Verification Cases and Procedures	A new verification procedure will be developed to ensure proper implementation of the software change.
16.15. Software Verification Results	The results of the new verification procedure will be included in the Software Verification Results.
16.16. Verification traceability	The new verification procedure will contain traceability to the currently existing and unchanged requirements.
16.17. Software Life Cycle Environment Configuration Index	Not applicable to DO-178A
16.18. Software Configuration Index	SCI will indicate new, modified, and re-used software life cycle data.
16.19. Problem Reports	No affect

16.20. Software Configuration Management Records	SCM records will track configuration item updates related to this change.
16.21. Software Quality Assurance Records	SQA records will track verification activities related to this change.
16.22. Software Accomplishment Summary	SAS will document the software life cycle activities performed in support of this change.
16.23. Tool Qualification Plan	Not applicable to DO-178A
16.24. Tool Operational Requirements	Not applicable to DO-178A
16.25. Tool Accomplishment Summary	Not applicable to DO-178A
16.26. Tool Verification Results	Not applicable to DO-178A
16.27. Other life cycle data (list)	Not applicable

Change Classification for SCR 1403

Aircraft Operational Safety Impact

The proposed software change will correct an existing problem related to improper flight plan leg transitions. Implementation of the proposed change will have a positive affect on aircraft operational safety.

Software Change Scope

The scope of the software change is limited to the addition of a single line of executable code. This new line of code is used to initialize a variable in lateral guidance to force determination of its proper value.

Change Classification

The change impact analysis performed on this proposed software change reveals that the software change is very limited in scope and, when implemented, will result in a positive affect on aircraft operational safety. Consequently, the proposed software change is classified as a minor software change. Further, in accordance with 14 CFR 21.611, the proposed change is classified as a minor change to the TSOA article.

Example Procedure
AlliedSignal

Appendix F



Software Modification Change Impact Analysis

QMS 5-04-303

Process Coordinator: Tom Roth

Process Leader: Dick Provencher

Printed versions of the Quality Management System documents are for reference only and are subject to change at any time. It is the responsibility of all employees to verify the hardcopy version against the electronic copy on the Intranet.

1 Introduction

This work instruction describes the proper use of the Request for Minor Software Modification Form QMS 4-04-303. The request form is the standard means for initiating, justifying and obtaining concurrence for a Minor Software Modification.

2 Definitions / Acronyms

ACO Aircraft Certification Office
CPU Computer Processor Unit
CTA Certified Test Article
FAA Federal Aviation Administration
FAI First Article Inspection
IPDS Integrated Product Delivery and Support
SCM Software Configuration Management
SQA Software Quality Assurance

3 References

QMS 5-04-300 EN - Embedded Software Process

4 Forms / Records

QMS 4-04-303 Minor Software Modification Form

5 Procedures

Minor software changes, like minor hardware changes, do not require CTAs or FAIs.

5.1 Responsibilities

The responsible engineer (IPDS Team Leader or Lead Software Engineer) is responsible for:

- completion of the Request for Minor Software Modification
- completion of the Justification for Minor Software Change

An SQA representative with DER credentials shall:

- review the Request for Minor Software Modification
- review Justification for Minor Software Modification
- determine Concurrence of Minor Software Modification

5.2 Minor Software Change Request

The requestor is to fill out the Request for Minor Software Modification and submit to SQA for signature. SQA determines the acceptance of change as candidate for Minor Software Modification.

5.3 Minor Software Change Justification

The requestor shall perform and validate the software modification as described on the Request for Minor Software Modification. The requestor shall then complete Justification of Minor Software Modification. The following sections describe the analysis required in the Justification portion of the form.

5.3.1 Traceability

Analysis performed to ensure requirements remain completely and accurately implemented and fully verified.

5.3.2 Memory Margin

Analysis performed to ensure memory allocation requirements are maintained.

5.3.3 Timing Margin

Analysis performed to ensure the timing requirements, CPU task scheduling requirements, system resource contention characteristics, or interface-timing requirements are maintained.

5.3.4 Data Flow

Analysis performed to identify changes to flow and coupling.

5.3.5 Control Flow

Analysis performed to identify changes to the flow and coupling.

5.3.6 Input / Output (Bus Loading)

Analysis performed to ensure that the change(s) have not impacted the I/O (bus loading) requirements established for the product.

5.3.7 Requirements and Design

Analysis performed to identify software requirements; software architecture and safety related software requirements impacted by the change

5.3.8 Code

Analysis performed to identify the software components impacted by the change.

5.3.9 Development Environment and Process

Analysis performed to identify change(s) that may adversely impact the software product (i.e. compiler options and optimization, tool qualification).

5.3.10 Documentation

Identify documentation requiring changes other than code..

5.3.11 Operational Characteristics

Identify change(s) which may affect operation characteristics (i.e. gain, filter and limit changes, sensed data validation, and fault annunciation).

5.3.12 Features / Functions Added

Identify all feature(s) or function(s) which are being added.

5.3.13 Testing Performed

Identify specific test procedures and cases that will be needed to be re-executed to verify the changes, new test procedures and cases (for added functionality or previously deficient testing) and that there is no adverse effects (regression testing).

5.3.14 Other

Identify any other areas of analysis that are appropriate and not mentioned above

5.4 Minor Software Change Concurrence

The responsible engineer for the product and the SQA representative will review the justification provided to support the position of Minor Software Modification. SQA shall determine concurrence that the justification supports a Minor Software Modification.

5.5 Minor Software Change Notification to the FAA

5.5.1 Initial notification of Minor Software Notification to the FAA

The SQA representative shall officially notify the FAA within a month of all the approved Minor Software Modifications within AlliedSignal Business & General Aviation. The completed Minor Software Modification forms will be submitted to the FAA ACO administrator.

5.5.2 Final notification of Minor Software Notification to the FAA

A formal submittal package must be sent to the FAA consisting of at a minimum the Software Configuration Index and Software Accomplishment Summary within one year of the concurrence of the Minor Software Modification or as a part of the next Major Software Modification which ever comes first. Any number of minor software modifications can be combined into a single FAA submittal provided they are all within the same one year time period. However, all minor software modifications must be accounted for in that submittal. The SQA representative reserves the right to force a formal submittal package at any time.

5.6 Minor Software Change Archival

The minor software modification form, Software Configuration Index, test procedures and test results shall be archived with the released software at the time of the release by the SCM group.

6 Flow Charts

See Appendix A

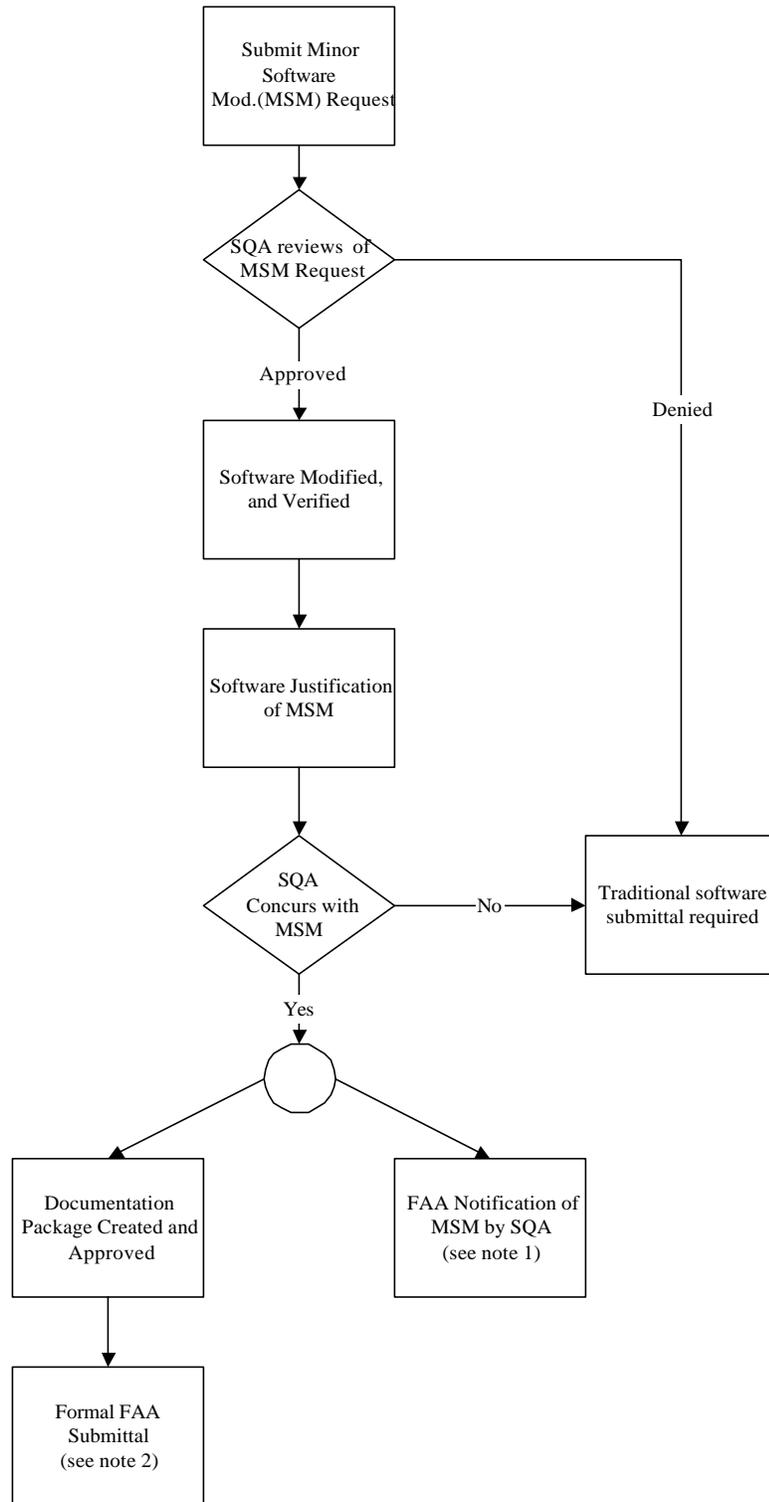
7 Revision History

Revision	Revision Date	Release Date	Description of Change
-		24 May 1999	Original release

Appendix A

Flow Charts

Minor Software Modification Flow Chart



Notes:

1. AlliedSignal notification to the FAA will occur with one month of SQA's concurrence.
2. Formal submittal required within 1 year of SQA's concurrence

Appendix B

Example of a Minor Software Modification Form

Request for Minor Software Modification

Requestor: Barry Webb
Date: 05/06/99
Unit Model: TPU 66A
TSO: C118
Unit P/N: 066-01145-0101
Current SW P/N: 206-00291-0111
New SW P/N: 206-00291-0112
SW MOD: 01/12
SW CID: 716-00120-0112

Complete Description of Change: The TPU 66A TCAS processor has an interface anomaly when connected to a competitor's newly released digital radar altimeter. The current TCAS software expects the altitude to trip to 4000 feet and provide SSM=NCD when the radar altimeter is out of range from the ground. The newly developed altimeter provides the actual value received, regardless of strength, and sets the SSM field to NCD when the return signal is too low. In this case the TCAS system will fail the altimeter. The failure only occurs at radar altitudes above 2500 but less than 4000 feet. The proposed resolution to the anomaly is for the TCAS software to be modified to consider the radar altimeter out of range from the ground whenever the SSM is set to NCD. This modification is already implemented in AlliedSignal's TCAS II system (TPU 67A processor with SW MOD 01/12).

Additionally, a modification is desired in testing the video board in the unit. A nuisance BITE failure can occur due to an inopportune task swap between two instructions during the board test (clearing a hardware bit and entering a MultiTask! critical region). The modification desired is to swap the two instructions to prevent this nuisance failure.

Responsible Engineer: _____

SQA Acceptance of Request: _____

Date: _____

Concurrence of Minor Software Modification

CO Number:

Responsible Engineer: _____

SQA Concurrence: _____

Date: _____

Justification of Minor Software Modification

Effect of Change on:	Analysis
Traceability	One requirement (para. 3.3.2.1.1.3.7.5 1d) and associated test (bbet_004) will have to be modified, but traceability will be unaffected.
Memory Margin	No impact
Timing Margin	No impact
Data Flow	No impact
Control Flow	No impact
Input / Output (bus loading)	No impact
Requirements and Design	Defect report #572 opened against the System Requirements Document removing the requirement to monitor radio altitude with SSM
Code	Module - i4hztask.c, function - rad_alt_update, delete 4 lines (remove check for > 4000 ft. when NCD and mark altitude as not credible and not invalid) Module - vbrdstst.c, function - video_board_md3x_bite, swap the order of two instructions
Development environment and process	No impact
Documentation	System requirement listed above under traceability
Operational characteristics	No impact
Identify features and functions being added	No impact
Identify test procedures	bbet_004 to validate radio altimeter modifications and MOPS high density tests 24321, 24332 and 2434 as regression tests for the system
Other (describe)	None

Change Impact Analysis	Yes	No
Previous hazards as identified by the system safety assessment are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Failure condition categories as identified by the system safety assessment are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Software levels are changed, especially if new level is A, B or C.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Safety related requirements as identified by the system safety assessment are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Software requirements, architecture and code components, especially those affecting safety related functions, partitioning, redundancy or safety monitors are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Code (source, object, executable) components that perform a critical function or a component which provides an input to a component which performs a critical function are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Characteristics of the development environment impacting the executable object code are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Memory allocation requirements are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Memory margin is adversely impacted.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Timing requirements are changed	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Timing margin is adversely impacted.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Bus loading (I/O) requirements are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Bus loading margin is adversely impacted.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Data and control coupling characteristics are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Protection mechanisms to ensure partitioning are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Processors or programmable devices are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Aircraft operational characteristics are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Flight crew procedures are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Flight Manual Supplement revision is required	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Aircraft worthiness characteristics are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Safety margins are reduced.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Pilot workload is increased.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Situational awareness, warnings and alerts are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Displayed information to make flight decisions is changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Assembly and installation requirements are changed.	<input type="checkbox"/>	<input checked="" type="checkbox"/>

Example Procedure
Honeywell Defense

Appendix G

WORK INSTRUCTION

Number:

WI8301211

Rev. - DRAFT

Date: February 25, 2000

Title: C-130J/382J/C-27J Procedure for Performing a Software Change Impact Analysis and Determination of Software Change Classification for Civil Certification Software

Approved: Steve Peterson

1 PURPOSE

This work instruction (WI) provides a procedure for performing a software change impact analysis and determination of a software change classification. Notice N8110.M&M describes the software change impact analysis process. The intent of a software change impact analysis is to:

1. Assure that aircraft operational safety is not **adversely** impacted due to a software change.
2. Determine the significance (scope) of the software change.
3. Provide justification for the change classification as it relates to 14 CFR 21.93 and 21.611.
4. Determine the extent of certification authority involvement in the review of software changes.

2 APPLICABILITY

The change impact analysis is applicable to software changes to be incorporated into a certified product (i.e., post certification software changes) on the C-130J/C-27J DA/FD and CNI projects. The change impact analysis should occur prior to the implementation of the software change. Consequently, the data derived from the change impact analysis may be only an estimation of the actual software change impact.

3 GENERAL

3.1

This work instruction is maintained by the C-130J/C-27J Quality Assurance Team and is reviewed/approved by the C-130J/C-27J System Review Board (SRB) (i.e. Configuration Manager, Quality Assurance Representative, Project Lead, and FAA DER).

This work instruction is archived and can be retrieved by the Information Technology team.

3.2

Appendix A contains the Software Change Impact Analysis Summary form.

3.3 ACRONYMS & DEFINITIONS

ACO	Aircraft Certification Office
CFR	Code of Federal Regulations
CNI	Communication Navigation Identification
DA/FD	Digital Autopilot / Flight Director

WORK INSTRUCTION

Number:

WI8301211

Rev. - DRAFT

Date: February 25, 2000

Title: C-130J/382J/C-27J Procedure for Performing a Software Change Impact Analysis and
Determination of Software Change Classification for Civil Certification Software

DER	Designated Engineering Representative
FAA	Federal Aviation Administration
PSAC	Plan for Software Aspects of Certification
SAS	Software Accomplishment Summary
SCI	Software Configuration Index
SRB	System Review Board
WI	Work Instruction
WIS	Work Instruction Standard

WORK INSTRUCTION

Number:

WI8301211

Rev. - DRAFT

Date: February 25, 2000

Title: C-130J/382J/C-27J Procedure for Performing a Software Change Impact Analysis and Determination of Software Change Classification for Civil Certification Software

4 IMPLEMENTATION

4.1 Change Impact Analysis Procedure Implementation

Step #	Responsible Party	Task
1	Systems Lead, Software Lead, Hardware Lead, Safety Engineer, Test Lead, and DER	<p>Complete Software Change Impact Analysis Form Prior to implementation of any post-certification software change, complete Table 1, "Software Change Impact Analysis Summary Form".</p>
2	Software Lead, Systems Lead, Safety Engineer, and DER	<p>Determine Software Change Classification Evaluate the data collected in Step 1 to:</p> <ol style="list-style-type: none"> 1. Determine if the change could adversely affect aircraft operational safety. 2. Determine the scope of the change. 3. Determine the change classification per the appropriate regulations (e.g., Part 21.93 and/or Part 21.611) and the minor/major change definitions provided in this WI. <p>For purposes of the change classification, the following definitions are used:</p> <p>Major change: Any change that could adversely affect aircraft operational safety.</p> <p>Minor change: Any change determined to have no adverse affect on aircraft operational safety or any change determined to have a positive affect on aircraft operational safety.</p>
3	DER	<p>Obtain Certification Authority Change Classification Concurrence For minor software changes</p> <ol style="list-style-type: none"> 1. Notify the certification authority of plans to implement a minor software change. Submit the following documentation to the certification authority: <ul style="list-style-type: none"> • Software Change Impact Analysis Summary • Assessment of change impact on aircraft operational safety • Assessment of the scope of the change • Justification of minor change classification 2. Obtain certification authority concurrence with the minor change classification.

WORK INSTRUCTION

Number:

WI8301211

Rev. - DRAFT

Date: February 25, 2000

Title: C-130J/382J/C-27J Procedure for Performing a Software Change Impact Analysis and Determination of Software Change Classification for Civil Certification Software

Step #	Responsible Party	Task
		<p><u>For major software changes</u></p> <ol style="list-style-type: none"> 1. Notify the certification authority of plans to implement a major software change. Submit the following documentation to the certification authority: <ul style="list-style-type: none"> • Software Change Impact Analysis Summary • Assessment of change impact on aircraft operational safety • Assessment of the scope of the change • Justification of major change classification • The planned life cycle activities to be performed in support of the change implementation <p>Note: Each of the items above can be included in a PSAC or can be a standalone document.</p> <ol style="list-style-type: none"> 2. Obtain certification authority concurrence with the major change classification.
4	Development Teams (Systems, Software, Test)	<p>Implement Software Changes</p> <p><u>For all software changes</u></p> <ol style="list-style-type: none"> 1. Proceed with implementation of the software change. <p>Note: Per N8110.M&M, equipment containing changes that are classified by the manufacturer as minor but not yet concurred with by the ACO or DER should be withheld from installation on flight aircraft until the ACO concurs with the classification.</p>
5	DER	<p>Submit Software Change Implementation Data to Certification Authority</p> <p><u>For minor software changes</u></p> <ol style="list-style-type: none"> 1. Document the software life cycle activities performed in support of the change implementation in the SAS. 2. Identify the new, modified, and re-used software life-cycle data in the SCI. 3. Submit the SAS and SCI to the certification authority per agreement with the certification authority. <p><u>For major software changes</u></p> <ol style="list-style-type: none"> 1. Document the software life cycle activities performed in support of the change implementation in the SAS.

WORK INSTRUCTION

Number:

WI8301211

Rev. - DRAFT

Date: February 25, 2000

Title: C-130J/382J/C-27J Procedure for Performing a Software Change Impact Analysis and
Determination of Software Change Classification for Civil Certification Software

Step #	Responsible Party	Task
		<ol style="list-style-type: none">2. Identify the new, modified, and re-used software life-cycle data in the SCI.3. Submit the SAS and SCI to the certification authority and/or DER.

WORK INSTRUCTION

Number:

WI8301211

Rev. - DRAFT

Date: February 25, 2000

Title: C-130J/382J/C-27J Procedure for Performing a Software Change Impact Analysis and Determination of Software Change Classification for Civil Certification Software

5 FORMS

Appendix A contains the Software Change Impact Analysis Summary form.

6 REFERENCES

FAA Notice N8110.M&M	Guidelines for the Oversight of Software Change undated Analyses Used to Classify Software Changes as Major or Minor	(Draft submitted for signature)
AC 25-19	Certification Maintenance Requirements	28 November 1994

Appendix A

Software Change Impact Analysis Summary Form

1. Identify the aircraft on which the product is installed. [For TSO articles, identify the intended target aircraft environment].	
2. Describe why the software change is needed.	
3. Describe the software change to be implemented.	
4. Requirements/design/code traceability analysis Note 1: Items that are affected include items that are changed, added, or deleted. Note 2: Safety related requirements/design/code are those related to monitoring, redundancy, protection/partitioning, etc.	
4.1. Identify the affected requirements in the SRD and SDD	
4.2. Identify the affected architectural design	
4.3. Identify the software components (procedures/functions) to be changed	
4.4. Identify any affects to safety related requirements	
4.5. Identify any affects to safety related architectural design	
4.6. Identify any affects to safety related procedures/functions	
4.7. Identify any new features or functions to be implemented	
4.8. Identify any affects to hardware design	
5. Verification traceability analysis Note 1: Verification activities can include review, analysis, regression testing, re-execution of existing tests, requirements-based testing, structural testing, etc.	
5.1. Identify the affected (i.e., changed, added, or deleted) verification procedures including any new procedures necessary for new features/functions or previously deficient verification	
5.2. Identify the verification activities which will be performed to ensure that: <ul style="list-style-type: none"> • the changes are verified • the changed software complies with product requirements • there are no adverse effects as a result of the change • new features or functions are verified • new features or functions do not adversely impact existing features or functions 	
6. Memory margin analysis	
6.1. Estimate the change to the flash memory	
6.2. Estimate the change to the RAM memory	
6.3. Determine if acceptable memory allocation margins are maintained (e.g., memory margin would be adversely affected if there is less than 5% margin remaining)	
7. Timing margin analysis	
7.1. Estimate the change to throughput margin for each affected task	
7.2. Determine if acceptable timing allocation margins are maintained (e.g., timing margin would be adversely affected if it is unpredictable or there is less than 10% margin remaining)	
8. Data coupling analysis	
8.1. Analyze data coupling changes to identify software components (procedures/functions) which would be affected by the proposed software change	

8.2.	Determine if data coupling changes have resulted in any adverse impacts (e.g., to the extent that more than 50% of the coverage analysis must be redone)	
9. Control coupling analysis		
9.1.	Analyze control coupling changes to identify software components (procedures/functions) which would be affected by the proposed software change	
9.2.	Determine if control coupling changes have resulted in any adverse impacts (e.g., to the extent that more than 50% of the coverage analysis must be redone)	
10. External input/output interface analysis		
10.1.	Identify changes to the external interfaces (e.g., databus I/O, hardwired I/O, etc.)	
10.2.	Determine if I/O changes have resulted in any adverse impacts (e.g., I/O margin would be adversely affected if there is less than 5% margin remaining)	
11. Hardware component analysis		
11.1.	Identify processor or interface hardware changes	
11.2.	Determine if hardware component changes have resulted in any adverse safety impacts	
12. Development environment analysis		
12.1.	Identify changes to development tools	
12.2.	Identify changes to the development environment (e.g., build process including compiler versions, optimization, options; linker, assembler, loader instructions)	
13. Software operational characteristics analysis		
13.1.	Identify changes to gains, filters, limits, data validation, interrupt and exception handling, and fault mitigation	
13.2.	Determine if software operational characteristic changes have resulted in any adverse impacts	
14. Safety analysis		
14.1.	Identify affected failure conditions, severity classifications, and probability of occurrence	
14.2.	Identify affected software criticality levels	
15. Aircraft operational analysis		
15.1.	Identify affects on the aircraft operational requirements or limitations	
15.2.	Identify affects on airworthiness characteristics	
15.3.	Identify affects on flight crew procedures and workload	
15.4.	Identify affects on situational awareness including display of cockpit information	
15.5.	Identify affects on flight crew advisories, cautions, and warnings	
15.6.	Identify affects on installation requirements or limitations	
15.7.	Identify affects on equipment interchangeability and/or interoperability with other equipment	
15.8.	Identify affects on certification maintenance requirements (CMR)	
16. Life cycle data analysis		
16.1.	Plan for Software Aspects of Certification	
16.2.	Software Development Plan	

16.3. Software Verification Plan	
16.4. Software Configuration Management Plan	
16.5. Software Quality Assurance Plan	
16.6. Software Requirements Standards	
16.7. Software Design Standards	
16.8. Software Code Standards	
16.9. Software Requirements Data	
16.10. Software Design Description	
16.11. Source Code	
16.12. Executable Object Code	
16.13. Requirements traceability	
16.14. Software Verification Cases and Procedures	
16.15. Software Verification Results	
16.16. Verification traceability	
16.17. Software Life Cycle Environment Configuration Index	
16.18. Software Configuration Index	
16.19. Problem Reports	
16.20. Software Configuration Management Records	
16.21. Software Quality Assurance Records	
16.22. Software Accomplishment Summary	
16.23. Tool Qualification Plan	
16.24. Tool Operational Requirements	
16.25. Tool Accomplishment Summary	
16.26. Tool Verification Results	
16.27. Other life cycle data (list)	

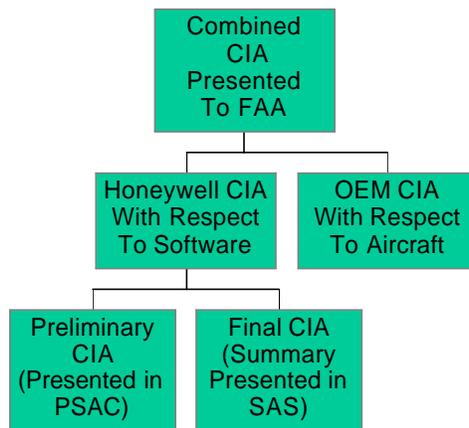
Example Process Presentation
Honeywell Phoenix

Appendix H

The CIA Process and Outputs

DRAFT

Overview of the CIA Process



Preliminary CIA in PSAC

- Contains a description of a single change or groupings of changes proposed for a certification
- Contains a major -vs- minor classification with a justification of the classification

Preliminary CIA Example 1

- Change Description (FDR)
 - Changes will be implemented to comply with the FAA mandate that additional aircraft parameters be recorded by the Flight Data Recorder (FDR) system. The IC-600 and DA-800 will provide several additional parameters that must be recorded by the FDR. As part of this change a new cautionary CAS message “FDAU FAIL” will be added to indicate that the Flight Data Acquisition Unit (FDAU) has failed.

Preliminary CIA Example 1 (cont.)

- Determination as Major or Minor Change with respect to software (FDR)
 - The FDR related changes are classified as minor.
 - The new data outputs already exist in the P-1000 system and this data will be provided on existing data busses.
 - The change for the CAS message is limited in scope because the CAS messaging system was designed to ensure logic partitioning between the messages. The new input required to drive the message will be read from an existing ARINC 429 data bus and the data flow is limited to this message.

Preliminary CIA Example 2

- Change Description (DAU Software Identifiers)
 - Changes will be implemented to display the Data Acquisition Unit (DAU) software identifiers in the IC-600 maintenance pages. This change will be implemented as part of the FLS initiative for the P-1000 system. Implementing this change will provide maintenance personnel with a convenient method to verify the DAU software identifiers following a software update. The display of the DAU software identifiers will be similar to the existing display of the IC-600 part numbers. The DA-800 will transmit the software identifiers on an existing ARINC 429 data bus. The IC-600 will be changed to provide a new on-ground maintenance page to display the software identifiers.

Preliminary CIA Example 2 (cont.)

- Determination as Major or Minor Change with respect to software (DAU SW Identifiers)
 - The change to display the DAU software identifiers is classified as minor. This change will be limited because the software identifier data is already contained in the DAU and will now be provided on an existing ARINC 429 data bus. The data will only be used for display purposes. The display of this data is also limited because it will be displayed in the on-ground maintenance pages. The maintenance pages are not accessible in-flight.

Preliminary CIA Example 3

- Change Description (CAT2 Logic)
 - A change will be implemented to the CAT 2 logic. The existing P-1000 logic only requires the coupled FD radio altitude to be displayed in order to enable the CAT 2 monitors and annunciations. The logic will be changed to require that both radio altimeter displays be valid before the CAT 2 monitors and annunciations are enabled.
 - Determination as Major or Minor Change with respect to software
 - The CAT 2 logic change is classified as minor. This will be a simple logic change which uses existing input variables.

Preliminary CIA Example 4

- Change Description (Upper Fuel Display Limit)
 - A change will be implemented to the MFD system pages to make the upper fuel limit of the analog fuel display scalable based on Long Range (LR) configuration strap. This change will allow the display scale indication of the fuel tanks to appear full when the fuel tanks are actually full. Previously the system used a fixed upper fuel limit for the analog display and on aircraft with smaller fuel tanks (ER version) the analog display would indicate less than full when the fuel tank was actually full.

Preliminary CIA Example 4 (cont.)

- Determination as Major or Minor Change with respect to software (Upper Fuel Display Limit)
 - The fuel display limit change is classified as minor. This change uses configuration strap data which is already contained in the system (LR strap). The change will allow a different constant to be used for the fuel scale limit based on the existing of LR configuration strap.

Final CIA

- Contains two parts
 - Supporting documentation found in outputs of development process
 - Summary of CIA in SAS

Final CIA Contents and where they may be documented in Development process

- Description of changes
 - Found in configuration management tool, PSAC, and SAS
- Effect on Documentation
 - Found in Version Description Document and Configuration management tool
- Timing and Memory impacts in SAS

Final CIA Contents and where they may be documented in Development process (cont.)

- Files effected
 - Found in configuration management tool and CAFs
- Testing impact
 - Found in configuration management tool and TPPR

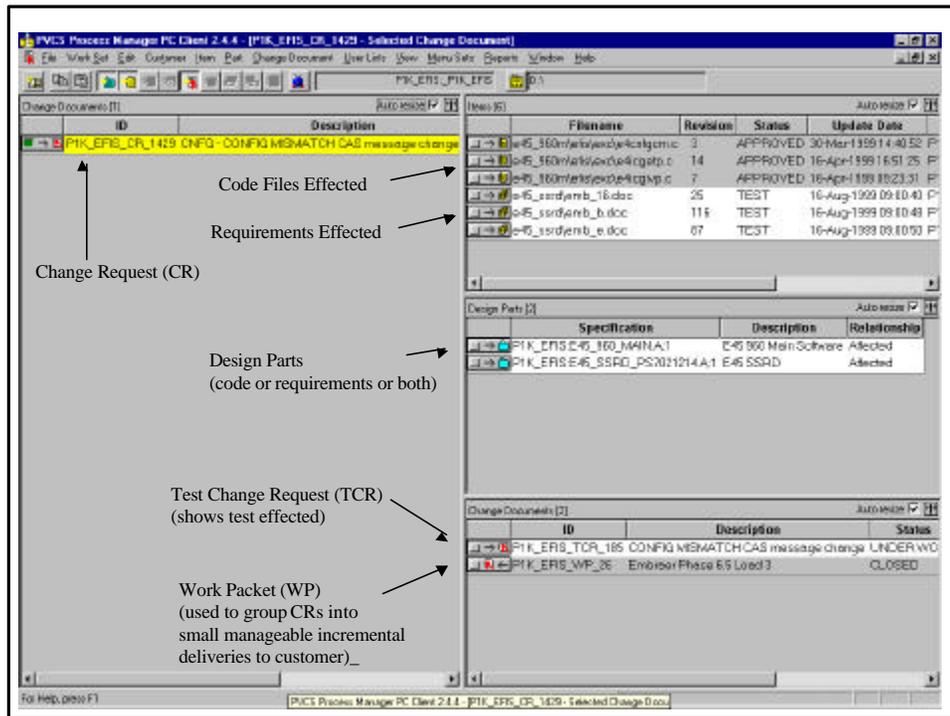
Final CIA Contents and where they may be documented in Development process (cont.)

- Data flow and Control analysis
 - checklist item during code review to look for erroneous side effects caused by the change as well as additional test to be run

Examples of CIA documentation
as a result of development
process

PCMS (our configuration management
system) Change Request (CR) View on
the next slide Shows:

- Design parts effected
- Requirement files effected
- Code files effected
- Related test change request (TCR)
- Related work packet (WP)



A CR Browsed (next slide) in PCMS Shows:

- Change Description
- Attributes
 - CR Priority
 - Problem/change Category
 - CR Source
 - Planned Delivery Information

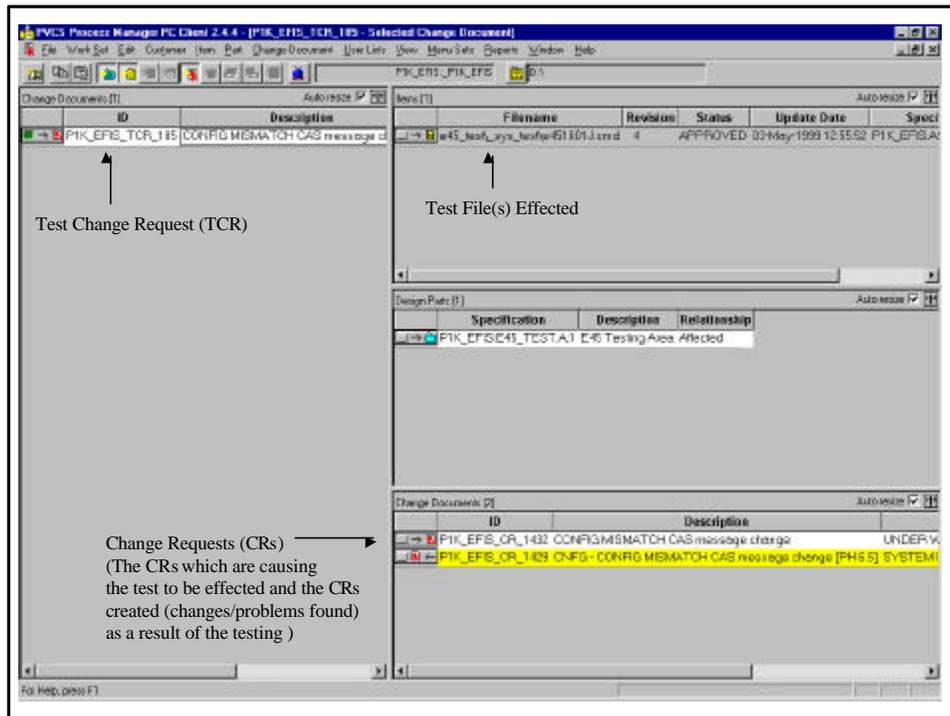
```
pt1392 - Notepad
File Edit Search Help

CHANGE REQUEST (CR)      CR Number:  P1K_EFIS_CR_1429

                          PROJECT: P1K_EFIS
                          APPLICATION: E45
                          FORMER PCR NUMBER (IN ACCE): S25-00000
-----
TITLE:  CNFG - CONFIG MISMATCH CAS message change [PH 6.5]
-----
ORIGINATOR:  Ken Hurt          PHONE: 4061
CREATED DATE: 11-AUG-1999     CURRENT STATUS: SYSTEM INTEGRATION TEST
-----
DESCRIPTION:
Change CONFIG MISMATCH CAS message per ECM1260 for Phase 6.5.
Items Aircraft ID, Engine Type, Long Range Version, and English Metric in
Table 16-6 are to flag an amber CONFIG MISMATCH while remaining items
in the table are to remain connected to an cyan CONFIG MISMATCH.
-----
                          PRIORITY: 2 - Customer Requirement
                          PROBLEM CATEGORY: P - Appl/Cust Requirement
                          FOUND IN PHASE: 8 - Customer
                          CR SOURCE: ECM 1260
                          TARGETED FUNCTIONAL DELIVERY: Load 3
                          TARGETED RELEASE: Phase 6.5
-----
EFFORT EXPENDED (in PERSON DAYS)
SYSTEMS ESTIMATED: 4      SOFTWARE ESTIMATED: 5
SYSTEMS ACTUAL:          SOFTWARE ACTUAL:
-----
HIGHER LEVEL (PARENT) RELATED CHANGE DOCUMENTS (X-REFs):
Cross Reference P1K_EFIS_MP_26 CLOSED Tom Nischnick (NISCHNICK)
```

PCMS TCR (test change request) View Shows:

- Test files effected
- The CR which caused the test files change
- CRs which where generated as a result of running the test.



Course Evaluation Forms

Appendix I

There are two course evaluation forms in this Appendix. Please select the one appropriate for your method of study.

- IVT broadcast
- Self-study video course

If you are taking this course via IVT and you are logged on to a keypad, you will be asked to complete the course evaluation by using the Viewer Response System keypad. Your IVT instructor will provide directions on how to complete the course evaluation. If you do not have access to a keypad, circle your responses and fax the *IVT Course Evaluation Form* to the IVT studio.

If you have completed this course by watching the video, please complete the *Self-Study Video Evaluation Form* and return to your directorate/division training manager (ATM). Your ATM must have your evaluation form in order to issue you credit for the course.

IVT COURSE EVALUATION

Software Change Impact Analysis

May 11, 2000

Please give us your candid opinions concerning the training you've just completed. Your evaluation of the IVT course is important to us, and will help us provide the best possible products and services to you. **NOTE: Your keypad responses are not identifiable by name; only average item responses are provided to the instructor and to others responsible for the training.**

Use your Viewer Response keypad to answer the following questions.

	Very Good	Good	Average	Poor	Very Poor
1. Length of course	A	B	C	D	E
2. Depth of information	A	B	C	D	E
3. Pace of training	A	B	C	D	E
4. Clarity of objectives	A	B	C	D	E
5. Sequence of content	A	B	C	D	E
6. Quality of course materials	A	B	C	D	E
7. Quality of graphics/visual aids	A	B	C	D	E
8. Readability of text on monitor	A	B	C	D	E

	Very Good	Good	Average	Poor	Very Poor
9. Effectiveness of instructor(s)	A	B	C	D	E
10. Communication between student and instructor	A	B	C	D	E
11. Applicability of material to your job	A	B	C	D	E
12. Overall quality of the course	A	B	C	D	E
13. Overall effectiveness of the IVT format	A	B	C	D	E
14. Would you like to take other IVT courses?	A. YES B. NO C. UNDECIDED				
15. On the keypad, enter your number of years of FAA experience.	_____ (number/enter)				

When finished, press the “Next Quest” key on your keypad and answer YES, then ENTER. Your responses will be sent electronically. Individual responses are not tabulated; only item averages for each question are presented to the instructor(s) and to AIR-510.

Additional Comments may be faxed to the IVT Studio:

405-954-0317 / 9507

SELF-STUDY VIDEO EVALUATION

Please give us your candid opinions concerning the training you've just completed. Your evaluation of the self-study video course is important to us, and will help us provide the best possible products and services to you.

Course title: _____

Date: _____

Number of years of FAA experience: _____

(Optional)Name: _____ Office phone: () _____

For the following, please darken the circle appropriate to your response.

	Very Good	Good	Average	Poor	Very Poor	N/A
1. Length of course	<input type="radio"/>					
2. Depth of information	<input type="radio"/>					
3. Pace of training	<input type="radio"/>					
4. Clarity of objectives	<input type="radio"/>					
5. Sequence of content	<input type="radio"/>					
6. Amount of activities/practice	<input type="radio"/>					
7. Quality of course materials	<input type="radio"/>					
8. Effectiveness of instructor(s)	<input type="radio"/>					
9. Overall quality of the course	<input type="radio"/>					
10. Overall effectiveness of the self-study video format	<input type="radio"/>					

11. Rate your level of knowledge of the topic before and after taking this self-study course.

	Very Low	Low	Moderate	High	Very High
BEFORE THE COURSE:	<input type="radio"/>				
AFTER THE COURSE:	<input type="radio"/>				

12. What did you like best about the course?

13. What would you improve in the course?

14. What previous experience, if any, have you had with self-study courses?

None Moderate Considerable

15. Were you comfortable with the self-study video format?

Yes No Undecided

If not, why not?

16. Would you like to take other self-study video courses?

Yes No Undecided

If not, why not?

17. Additional comments:

**PLEASE SEND THIS COMPLETED FORM TO YOUR
DIRECTORATE/DIVISION TRAINING MANAGER (ATM). THANK YOU.**