DOT/FAA/AR-01/125

Software Service History Report

Office of Aviation Research
Washington, D.C. 20591

January 2002

Final Report

U.S. Department of Transportation
Federal Aviation Administration

NOTICE

This report is available at the Federal Aviation Administration William J. Hughes Technical Center's Full-Text Technical Reports page: actlibrary.tc.faa.gov in Adobe Acrobat portable document format (PDF).

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
DOT/FAA/AR-01/125		

4. Title and Subtitle	5. Report Date
SOFTWARE SERVICE HISTORY REPORT	January 2002
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Uma D. Ferrell and Thomas K. Ferrell	

9. Performing Organization Name and Address	10. Work Unit No. (TRAIS)
Ferrell and Associates Consulting, Inc. 1261 Cobble Pond Way Vienna, VA 22182	11. Contract or Grant No. DTFA0300P10138

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
U.S. Department of Transportation Federal Aviation Administration Office of Aviation Research Washington, D.C. 20591	Final Report
	14. Sponsoring Agency Code AIR-130

15. Supplementary Notes

This report is one of two deliverables under this contract. The first being the Software Service History Handbook. The FAA William J. Hughes COTR is Charles Kilgore.

16. Abstract

The safe and reliable operation of software within civil aviation systems and equipment has historically been assured through the application of rigorous design assurance applied during the software development process. Increasingly, manufacturers are seeking ways to use software that has been previously developed for other domains or that has been previously certified for use in lower criticality aviation applications. Product service history is one method for demonstrating that such software is acceptable for use in the new application domain. In theory, product service history would seem to be a fairly simple concept, both to understand and to apply. However, in practice, such use has proved extremely problematic, as questions of how to measure the historic performance and the relevance of the provided data have surfaced. This report represents the results of research in this area, for use by the Federal Aviation Administration in formulating new guidance and prioritizing future research work in the area of product service history.

17. Key Words	18. Distribution Statement
DO-178B, SC-190, DO-248, Product service history, Software reliability, Airborne systems and equipment, Error rates	This document is available to the public through the National Technical Information Service (NTIS) Springfield, Virginia 22161.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	88	

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AC	Advisory Circular
ACO	Aircraft Certification Office
ATM	Air Traffic Management
CAST	Certification Authorities Software Team
CNS	Communication, Navigation, Surveillance
COTR	Contracting Officer's Technical Representative
COTS	Commercial Off-The-Shelf
DO	Document
DOT	Department of Transportation
ED	European Document
EUROCAE	European Organization for Civil Aviation Equipment
FAA	Federal Aviation Administration
FAR	Federal Aviation Regulations
HBAW	Airworthiness Handbook
ICSE	International Conference on Software Engineering
IEEE	Institute of Electrical and Electronic Engineers
ISO/IEC	International Organization for Standardization/International Electrotechnical Commission
JAA	Joint Airworthiness Authorities
MTTR	Mean Time to Repair
MoD	Ministry of Defense, United Kingdom
NASA	National Aeronautics and Space Administration
OSED	Operational Services and Environment Definition
PDF	Portable Document Format
PDS	Previously Developed Software
PSAC	Plan for Software Aspects of Certification
RTCA	formerly Radio Technical Commission for Aeronautics
SAS	Software Accomplishment Summary
SC	Special Committee
SCM	Software Configuration Management
SOUP	Software of Unknown Pedigree
SSA	System Safety Assessment
SSAC	Streamlining Software Aspects of Certification
TGL	Temporary Guidance Leaflet

EXECUTIVE SUMMARY

The safe and reliable operation of software within civil aviation systems and equipment has historically been assured through the application of rigorous design assurance applied during the software development process. Increasingly, manufacturers are seeking ways to use software that has been previously developed for other domains, has been previously certified for use in lower criticality aviation applications, or has been certified to earlier versions or different standards than those currently employed. Product service history is one method for demonstrating that such software is acceptable for use in a new application. In theory, product service history would seem to be a fairly simple concept, both to understand and to apply. However, in practice, such use has proved extremely problematic, as questions of how to measure the historic performance and the relevance of the provided data have surfaced. To date, no specific guidance has been produced to aid in the formulation of service history approaches beyond the limited discussion in DO-178B, "Software Considerations in Airborne Systems and Equipment Certification." This research effort is designed to collect, analyze, and synthesize what is known and understood about evaluating product service history.

Two deliverables were produced as part of this research effort: a handbook and this report. The handbook, the contents of which are included within this report, is intended for use by both industry and the Federal Aviation Administration (FAA) in applying and evaluating service history arguments made for the purposes of obtaining certification credit. The handbook was limited in scope to cover only those items for which there is current supporting guidance within DO-178B. This technical report presents the results of this research effort for use by the FAA in formulating new guidance and prioritizing future research work in the area of product service history.

Using a taxonomy of questions derived from the definition of product service history in DO-178B, both quantitative and qualitative considerations are explored. This discussion is extended by inclusion of additional questions from other industries in which service history is used in evaluating software maturity. A set of worksheets is included for evaluating the relevance and sufficiency of service history data for possible certification credit.

Further extending the above taxonomy, gaps in the current guidance relative to each area of questions and the associated paragraphs in DO-178B are highlighted along with possible approaches to close these gaps. Following the discussion of the four parts of the product service history definition, a description of the relationship of process assurance and equivalent safety to product service history is provided. The report concludes with a research summary and suggestions for further actions stemming from this research effort.

1. INTRODUCTION.

1.1 PURPOSE.

This report, produced under a Federal Aviation Administration (FAA) research contract, discusses gaps in industry practice and knowledge of software service history for which further work is needed. It is closely related to the Service History Handbook that was also created as part of this research effort. This report is intended for use as a planning document for policy and guidance development, as well as follow-on research efforts in the areas of service history credit for certification and system approvals. Both the report and the handbook adopt the same questions-based approach to highlight specific areas of concern in the evaluation of service history data.

The primary subject of both the handbook and report is the provision of certification credit for software for which little is known about the design assurance and verification activities. Data from the development may be incomplete or inaccessible to those wishing to use the software in a safety-related application. Of particular concern are the differences between formal verification of the software and experience realized through service usage of the software.

While the handbook is intended for use by both the FAA and industry practitioners, this report is primarily for FAA use. It expands on a number of topics found in the handbook for the purposes of highlighting specific gaps and contradictions that were found during the course of the research effort. It also serves as the collection point for the raw data that was collected and reviewed during the execution of the service history contract. For completeness, the technical content of the handbook also appears in this document. This allows the report to stand alone.

1.2 SCOPE.

The scope of this report and the underlying research is restricted to software used in airborne applications. This would include avionics, electronic engine controls, in-flight entertainment systems, etc. While aviation-related use of software service history on the ground was not included, it is expected that many of the same issues apply.

While primarily focused in the area of software, it should be noted that many of the concerns and issues raised in the report might also be extended for application to complex electronic hardware.

1.3 BACKGROUND.

During the creation of DO-178B, product service history was identified as a possible alternative method for demonstrating compliance to one or more of the objectives in DO-178B[1]. To date, use of this method has been limited due to both the difficulty in demonstrating the relevance and sufficiency of the product service history, as well as a lack of any consistent guidance for approaching such a demonstration. It is hoped that the results of this research may allow such guidance to be developed.

[1] DO-178B is invoked by the FAA through Advisory Circular 20-115B. ED-12B, the EUROCAE equivalent of DO-178B, is similarly invoked by Temporary Guidance Leaflet (TGL) #4, by the Joint Airworthiness Authorities (JAA).

This report, along with the accompanying handbook, attempts to capture in one place what is known about the topic of product service history. Using the guidance provided in DO-178B as a starting point, other safety-critical industries were canvassed in an attempt to identify if service history was being used as part of system evaluations, and if so, in what manner. Similarly, other sources of guidance for the aviation industry were explored, most notably the work accomplished by Radio Technical Commission for Aeronautics (RTCA) Special Committees 180 and 190 (SC-180 and SC-190) and by the Certification Authorities Software Team (CAST).

SC-180, which produced DO-254, "Design Assurance Guidance for Airborne Electronic Hardware," outlines how product service experience may be used "to substantiate design assurance for previously developed hardware and COTS components." DO-254 was released in April 2000. DO-254's treatment of product service experience is contained in two separate sections, 11.3 and Appendix B, paragraph 3.2. The guidance in 11.3 closely parallels the guidance in DO-178B for product service history. However, the guidance in the appendix requires additional design assurance for level A and B hardware if service experience is claimed. There is also a requirement to link any analysis of product service history experience into the functional failure path analysis for levels A and B. This is analogous to the tie back to system safety objectives required in 12.3.5 of DO-178B.

SC-190 is still an active committee, although their final work products are currently moving through editorial review as of the publication of this handbook. Their outputs include DO-248B and a guidance document for nonairborne Communication, Navigation, Surveillance/Air Traffic Management (CNS/ATM) ground systems. Within DO-248B, frequently asked question No. 19 and discussion paper No. 4 specifically address the use of product service history. The content of these items have been reflected in this handbook. Considerable discussion occurred in the SC-190 CNS/ATM subgroup on the establishment of a tiered approach to minimum service history duration based on criticality levels. No consensus could be reached on the minimum duration since the proposals were derived from the software reliability field that is not viewed by some to be a mature field.

CAST is comprised of representatives of certification authorities from Europe, Canada, and the United States. CAST meets regularly to discuss technical and regulatory matters related to the uniform interpretation of DO-178B and related guidance. CAST produced a position paper on the subject of product service history. Both the final paper and a number of earlier drafts were considered in the course of completing this research effort.

In addition to the aviation sources mentioned above, numerous references were reviewed from the nuclear, military, consumer products, and medical devices domains, as well as general software literature. The most mature treatment of the topics were found in Europe, most notably in standards published by the United Kingdom Ministry of Defense (MoD) and safety-consulting called Adelard. In addition to the written materials that were reviewed, a series of interviews were conducted with practitioners in these other fields to further explore exactly how the subject of service history was addressed in practice.

The final activity, prior to the actual creation of this handbook, was the conduct of a detailed breakout session during the FAA's National Software Conference held in Boston, MA, in June

2001. Preliminary results of the study were shared and feedback was sought relating to specific issues arising from the product service history definition. Both the interviews and the breakout session served to validate the findings from the literature review and contributed greatly to the final contents of both the report and handbook.

1.4 THE RESEARCH EFFORT.

The research effort consisted of four basic steps: data collection, data analysis, data synthesis (including gap analysis), and gap closure plan development. Two deliverables were produced as a result of these activities, a handbook usable by both the government and industry in evaluating and applying the concept of product service history and a report (this document) capturing the overall results of the research effort along with any identified gaps in existing guidance and plans for closing those gaps.

Figure 1 illustrates the basic flow for the four activities.

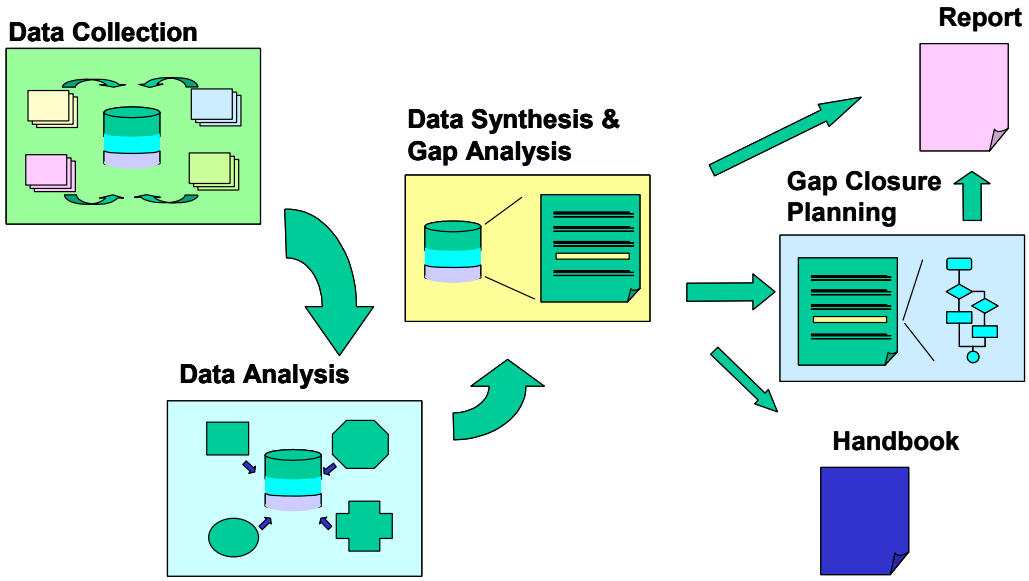

FIGURE 1. OVERALL RESEARCH EFFORT DESIGN

Two taxonomies (see figure 2) were identified early on as a way to organize the research effort. The first of these involved the elements of the product service history definition.

This taxonomy resulted in four specific areas of questions that were explored in depth. These were time, operations, environment, and problem reporting. The second taxonomy was intended to segregate the issues associated with evaluating product service history into regulatory, procedural, and technical components. While the first of these taxonomies worked well, the second was never fully realized. In reality, there is very little distinction that can be made between procedural and regulatory aspects of product service history. There is some value in separating out the technical aspects, which is accomplished in the research summary.

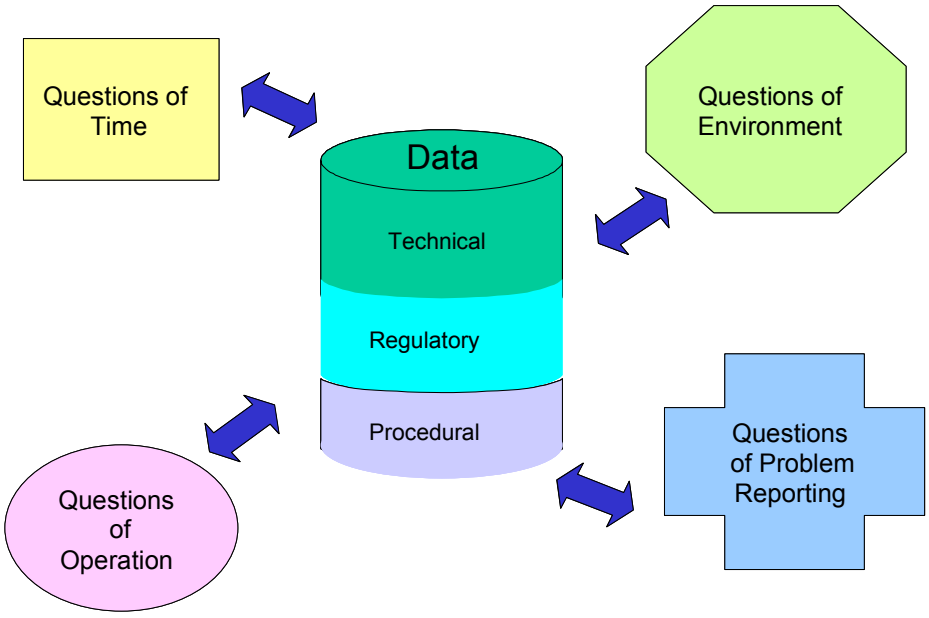

FIGURE 2. RESEARCH TAXONOMIES

Data collected in the first phase of the effort was analyzed and parsed using the taxonomy just discussed along with a series of questions related to the definition of product service history. A generic checklist, included as part of appendix A, was used to facilitate the data review and analysis process. The results of the data analysis were then used to populate the handbook. Where gaps were identified in the data, relating to how product service history could be evaluated, the gap was described and a possible approach to filling the gap was identified. This information was captured in the report as part of the data synthesis activity. A small number of interviews and a breakout session at the 2001 National Software Standardization Conference were used to validate the findings from these efforts. Final cleanup and assembly of both the report and handbook resulted in additional insights that are captured herein as possible future research areas.

1.5 RELATED ACTIVITIES/DOCUMENTS.

The following documents relate directly to the issues addressed herein and define the nature of the problem studied in this evaluation:

a. DO-178B/ED-12B
b. DO-254/ED-80
c. DO-248B/ED-94B
d. FARs and associated AC's, most notably

- XX.1309
- XX.1301
- XX.901

where XX represents the aircraft type for example, 25.1309, 23.1309 etc.

In addition, FAA Notices 8110.85 and 8110.89 are relevant to this discussion. Finally, specific discussion papers presented at the RTCA SC-167 and SC-190 meetings were reviewed and considered. In many cases, these papers contained useful ideas that were not included in the final products of their associated committees due to a lack of consensus or the constraints placed on the committee.

1.6 DOCUMENT STRUCTURE.

This report follows a similar structure to that found in the handbook. The major discussions in sections 2 through 5 are repeated from the handbook and are annotated with additional notes. Two subsections are introduced for each element of section 3. The first of these subsections incorporates the relevant worksheet that addresses the question area and contradictions in the material related to that specific topic. The second subsection outlines gaps and steps that might be taken to close the gaps ranging from improved guidelines for use of product service history to follow-on research for identifying workable approaches. There are two appendices that capture the detailed work accomplished to develop both the handbook and this report. These include the full bibliography, research notes from the source review, and detailed notes from the interviews. A brief summary of the contents is provided here for reference.

Section 1 provides introductory material including the purpose, scope, background, and related documents.

Section 2 discusses DO-178B's treatment of the Product Service History alternative method, as well as its definition.

Section 3 provides a detailed discussion of the elements that comprise the product service history definition.

Section 4 discusses the relationship of product service history to both process and product assurance.

Section 5 draws the various aspects of a product service history argument together for the purposes of illustrating how an equivalent level of safety argument might be made using product service history data.

Section 6 provides a summary of the research effort, as well as a set of recommendations to address systemic gaps.

Appendix A provides notes on data collection and data synthesis associated with the various sources (references, interviews, and the industry feedback session at the FAA Nation Software Conference).

Appendix B is the complete list of sources used in the literature search.

Note: Throughout both the handbook and the report, the language and the philosophy of DO-178B are retained. For example, the vocabulary used in various domains of this research is different from that used in DO-178B. Words such as in-service history, field data,

operating experience, operational history, proven-in-use data, and item history, are used for product service history. A translation has been performed to maintain commonality of terms with those used in DO-178B.

2. DO-178B FRAMEWORK.

With almost 20 years of usage by the airborne software community, DO-178 is the measure by which all airborne software in safety-critical applications is measured in civil aviation. As such, it is important that studies, such as this one, relate their findings back to DO-178B, and in particular, the processes identified in this guidance. Toward that end, every effort has been made to ensure that where gaps have been identified, the affected process and the corresponding objective evidence needed to demonstrate compliance with the process have been highlighted. Each proposal to fill the perceived gap has been placed in context of one or more DO-178B processes that would be satisfied once the gap is closed.

DO-178B outlines a total of 66 objectives that should be satisfied for software with the highest potential impact on safety in the event of its failure (Level A). Four additional levels of software are provided (Levels B through E), each with a decreasing number of objectives that must be satisfied as the potential safety impact is reduced. All of the objectives are described in the context of the software development process and a series of integral processes that cut across the entire software development effort. In addition, DO-178B discusses a small number of alternative methods for demonstrating compliance to 1 or more of the 66 objectives. Product service history is one such alternative method.

Note: The idea of software levels that correspond with specific levels of safety is maintained in both the handbook and the report. This tiered approach is common in other domains; however, the exact level definition may be slightly different. A translation was attempted for the purposes of comparison with and subsequent incorporation of specific approaches from these sources. However, difficulties in establishing system safety levels without using error rates prevented a recommendation of a tiered approach to product service history assurance.

2.1 THE DEFINITION OF PRODUCT SERVICE HISTORY.

DO-178B defines product service history as “a contiguous period of time during which the software is operated within a known environment, and during which successive failures are recorded.” This definition has three major components:

- Problem reporting
- Environment
- Time

For the purposes of this report, the environment component has been subdivided into two pieces, one focused on external operations such as operating modes, people, and procedures; and the second focused on computing (hardware environment) aspects of the software. Likewise, the problem reporting component has been broadened to include all facets of configuration management as they relate to the use of product service history.

DO-178B, Section 12.3.5, states that the acceptability of any argument predicated on the use of product service history depends on six items:

- Configuration Management of the Software
- Effectiveness of Problem Reporting Activity
- Stability and Maturity of the Software
- Relevance of Product Service History Environment
- Actual Error Rates and Product Service History
- Impact of Modifications

A total of 11 specific guidance statements are then provided for the purposes of evaluating and demonstrating that these six items have been satisfied. These 11 items (a through k), are discussed in section 2.2 of this report in the context of one or more elements of the definition of product service history.

2.2 ANALYSIS OF PRODUCT SERVICE HISTORY IN DO-178B.

The following table was constructed as a result of analysis of current guidance given in DO-178B, Section 12.3.5. Each item in this section was studied to understand what questions must be asked to get pertinent information and what additional considerations are not discussed directly in DO-178B. The taxonomy described in section 1.4 of this document (problem reporting, operation, environment, and time) is used to classify the contents of DO-178B, Section 12.3.5. A preliminary set of questions was derived during this analysis; these questions were later improved as gaps were recognized. The completed sets of questions within each taxonomy appear as worksheets in section 3.

TABLE 1. ANALYSIS OF DO-178B, SECTION 12.3.5

DO-178B Section 12.3.5 Reference	Observations on DO-178B Section 12.3.5	Software Service History Questions	Question Category
a. The applicant should show that the software and associated evidence used to comply with system safety objectives have been under configuration management throughout the product service history.	If this evidence for safety objectives is missing or noncompliant, there is no work around. The service history data that is associated with software that is not configuration controlled should not be used as evidence since there is no confidence in the data or software. There is no basis for computing error rates.	1. Are the software versions tracked during the service history duration? 2. Are problem reports tracked with respect to particular versions of software? 3. Are problem reports associated with the solutions/patches and an analysis of change impact? 4. Is revision/change history maintained for different versions of the software? 5. Have change impact analyses been performed for changes?	Question of Problem Reporting
b. The applicant should show that the problem reporting during the product service history provides assurance that representative data is available and that in-service problems were reported and recorded and are retrievable.	If this evidence is unsatisfactory, the significance of available service history data is reduced. There are a number of issues regarding the integrity of problem reporting system: the vendors' self-interest in not publishing all of the problem data, not all of the data may be reported, not all of the data may be recorded. In case there are doubts as to whether "representative" data are available, the available data may be used for finding out if there were safety critical problems during the service history duration without taking credit for certification. Although this item is describing the elements of configuration control, the guidance does not directly state that the problem reports need to be under configuration control.	1. Were in-service problems reported? 2. Were all reported problems recorded? 3. Were these problem reports stored in a repository from which they can be retrieved? 4. Were in-service problems thoroughly analyzed and are those analyses included or appropriately referenced in the problem reports?	Question of Problem Reporting

TABLE 1. ANALYSIS OF DO-178B, SECTION 12.3.5 (Continued)

DO-178B Section 12.3.5 Reference	Observations on DO-178B Section 12.3.5	Software Service History Questions	Question Category
c. Configuration changes during the product service history should be identified and the effect analyzed to confirm the stability and maturity of the software. Uncontrolled changes to the Executable Object Code during the product service history may invalidate the use of product service history.	This point is also dealing with the integrity of the problem data, in terms of whether the software was under configuration control. Without configuration control, the problem reports cannot be associated with specific versions of the software. Uncontrolled changes to software also indicate deficiencies in the development process.	1. Is each problem report tracked with its status of whether it is fixed or open? 2. If a problem was fixed, is there a record of how the problem was fixed? 3. Is there a record of the new version of software with the new release after the problem was fixed? 4. Are there problems with no corresponding record of change in software version? 5. Does the change history show that the software is currently stable and mature?	Question of Problem Reporting
d. The intended software usage should be analyzed to show the relevance of the product service history.	If there are fundamental differences in the usage, the product cannot be used in the proposed environment and there will not be any need to use the service history. In some cases, additional code, in the form of a "wrapper" or "glue code," may be needed to integrate the existing software into the new operating environment. In this case, the effect of this additional code on the relevance of the service history should also be considered.	1. Is the intended software operation similar to the usage during the service history? (Its interface with the external "world," people, and procedures.) 2. Have the differences between service usage and proposed usage been analyzed?	Questions of Operation and Environment
e. If the operating environments of the existing and proposed applications differ, additional software verification should confirm compliance with the system safety objectives.	There is no guidance on what elements of the two operating environments need to be compared. There exists a dilemma in the case of multiple copies of the product being used in many diverse operating environments, service history in a dissimilar environment may actually give a larger coverage of input/output space and thus resulting in problem reports, which may not have been written up if the product was executed only in a similar environment.	1. If the input/output domains differ between service history duration and intended use, has there been an analysis of what functions are covered by the service history? 2. Is the hardware environment of service history and the target environment similar? 3. Is the product compatible with the target computer without	Question of Operation and Environment

TABLE 1. ANALYSIS OF DO-178B, SECTION 12.3.5 (Continued)

DO-178B Section 12.3.5 Reference	Observations on DO-178B Section 12.3.5	Software Service History Questions	Question Category
	This circumstance has a larger probability of weeding out latent errors, although the service history in a dissimilar environment may not be used for error rate calculations.	making modifications to the product software? 4. If the computer environments are different, are the differences verified (through analysis and/or testing)? 5. Are there differences in the operating modes in the new usage?	
f. The analysis of configuration changes and the product service history environment may require the use of software requirements and design data to confirm the applicability of the product service history environment.	This item is a companion to items c and e. The text in DO-178B does not claim that software requirements and design data will be needed for this analysis nor does it claim that these are the only sources of needed information. In fact, the required information may be present from looking at the problem reports, version dates, patch descriptions, and perhaps details of how the problem was fixed which might be present in the problem report logs/database. If this additional information is insufficient, then the requirements and design data may need to be reviewed.	1. Is the data needed to analyze the similarity of environment available? (Such data are not usually a part of problem data.) 2. If not, have the software requirements and design data been reviewed to support the service history claim?	Questions of operation, environment, and problem reporting
g. If the software is a subset of the software that was active during the service period, then analysis should confirm the equivalency of the new environment with the previous environment and determine those software components that were not executed during normal operation. Note: Additional verification may be needed to confirm compliance with the system safety objectives for those components.	This item does not discuss what happens to functions that are not going to be used in the proposed application. Assurance requirements for dead code or deactivated code in DO-178B may apply, depending upon the technique used to prevent the usage of these functions.	1. Are only some of the functions of the proposed application used in service usage? 2. Is there a gap analysis of functions that are needed in the proposed application but have not been used in the service duration?	Questions of Operation and Environment

TABLE 1. ANALYSIS OF DO-178B, SECTION 12.3.5 (Continued)

DO-178B Section 12.3.5 Reference	Observations on DO-178B Section 12.3.5	Software Service History Questions	Question Category
h. The problem report history should be analyzed to determine how safety-related problems occurred and which problems were corrected.	If safety-related problems are still pending, the analysis must show that the problem is not an issue for the intended application. This is generally a difficult issue since, if the operational environment is similar, the safety critical problem is of significance; if the operational environment is not similar, then the service history is not of consequence.	1. Are all problems within the problem report repository classified? 2. Are safety-related problems identified as such? 3. Can safety-related problems be retrieved? 4. Is there a record of which problems are fixed and which problems are left open? 5. Is there enough data after the last fix of safety-related problems to assess that the problem is solved, and that no new safety-related problems have surfaced? 6. Do open problem reports have any safety impact?	Question of Problem Reporting
i. Those problems that are indicative of an inadequate process, such as design or code errors, should be indicated separately from those whose cause are outside the scope of this document such as hardware or system requirements errors.	The level of detail that is required here for logging problem reports with their solutions and the source of errors may itself indicate good engineering practice. A lack of sufficient information may be indicative of an inadequate process.	1. Are the problem reports and their solutions classified to indicate how a fix was implemented? 2. Is it possible to separate those problems that caused a design or code correction? 3. Is it possible to separate the problem reports that were fixed in the hardware or change of requirements?	Questions of Problem Reporting and Environment
The data described above and these items should be specified in the Plan for Software Aspects of Certification: j. (1) Analysis of the relevance of the product service history environment	Analysis of the relevance of the product service history environment is discussed in items d, e, f, g, and i.	All questions on similarity of usage and operational environment associated with these items apply.	Questions of Operation and Environment

TABLE 1. ANALYSIS OF DO-178B, SECTION 12.3.5 (Continued)

DO-178B Section 12.3.5 Reference	Observations on DO-178B Section 12.3.5	Software Service History Questions	Question Category
The data described above and these items should be specified in the Plan for Software Aspects of Certification: j. (2) Length of Service period and rationale for calculating the number of hours of service, including factors such as operational modes, the number of independently operating copies in the installation and in service, and the definition "normal operation" and "normal operation time."	Item j (2) has a requirement for calculating the length of service history duration by taking into consideration the number of hours of service, similar operation, and the number of independently operating copies. However, the independence here is not fully applied. It is inferred that the independence in operation should result in statistically independent error rates. The definition of normal operation is also left up to the applicant. This definition should be consistent with the proposed use of the software.	1. What is the definition of service period? 2. Is the service period defined appropriate to the nature of software in question? 3. How many copies of the software are in use and being tracked for problems? 4. Are the input/output domains the same? 5. What was the criterion for evaluating the service period duration? 6. What is the definition of normal operation time? 7. Does the service period include normal and abnormal operating conditions? 8. Is the definition of normal operation and normal operation time appropriate to the product?	Question of Time, Environment, and Operation
The data described above and these items should be specified in the Plan for Software Aspects of Certification: j. (3) Definition of what was counted as an error and rationale for that definition.	Item j (3) is not referenced directly in any of the other items; there is a reference to classification of problems in item h where the classification does not go beyond whether the problem was safety related. Definition of errors here refers to how errors were defined in the service history data. It is important that this definition have an acceptable rationale, and also, that the definition should have been applied to all of the problems consistently throughout the service history duration.	1. Was there a procedure used to log the problem reports as errors? 2. What was the reasoning behind the contents of the procedure? 3. Is there evidence that use of this procedure was enforced and used consistently throughout the service history period?	Question of Problem Reporting

TABLE 1. ANALYSIS OF DO-178B, SECTION 12.3.5 (Continued)

DO-178B Section 12.3.5 Reference	Observations on DO-178B Section 12.3.5	Software Service History Questions	Question Category
The data described above and these items should be specified in the Plan for Software Aspects of Certification: j. (4) Proposed acceptable error rates and rationale for the product service history period in relation to the system safety and proposed error rates.	Acceptable methods for computing error rates are not stated in DO-178B. The requirement that this has to be forecast in relationship with the system safety is in direct conflict with the idea that software reliability measures cannot be used to justify safety, nor can these numbers be used in system safety analysis. The duration that is used in the equation for computing error rates can be varied. If data exists from many application environments but only a few of these environments are similar to the target environment, not all available errors or the duration can be used in the error rate computation. In that case, if there were noted safety problems in the applications that are not considered in the error rate, should we be concerned? Common sense dictates that we should be. However, if we do not use this data in the error rate, how can we consider these problems?	1. Do you have a proposed error rate, justifiable and appropriate for the level of safety of proposed usage, (before analyzing the service history data)? 2. How do you propose that this error rate be calculated (before analyzing the service history data)? 3. Is the error rate computation appropriate to the application in question? Note: (Error rates may be computed in a number of different ways, such as total errors divided: • by time duration • by number of execution cycles • by number of events such as landing • by flight hours • by flight distance • by total population operating time • by number of hours during which the product is expected to perform (Other definitions for error rates may be used depending upon the particular usage.)	Questions of Time, Operation, Environment, and Problem Reporting

TABLE 1. ANALYSIS OF DO-178B, SECTION 12.3.5 (Continued)

DO-178B Section 12.3.5 Reference	Observations on DO-178B Section 12.3.5	Software Service History Questions	Question Category
k. If the error rate is greater than that identified in the plan, these errors should be analyzed and the analyses reviewed with the certification authority.	Same issues as in item j (4) above exist for this item. Unless a method for using error rates in system safety assessment is established, this item cannot be resolved objectively. Also, there is no mention here of whether the errors were safety related or not. High error rates of nonsafety problems cannot have the same weight as the low error rates of safety problems. Although, item h deals with an analysis of safety problems and the problems that may not have been fixed, the analysis is not required to be used in the review of error rates. Error rates during normal operation within the service history duration are usually large soon after deployment of major corrections and releases. It is not clear whether the referenced error rates are over the span of the entire period of service history or a selected section such as in the last year of usage. Although impact of modifications is cited as one of the considerations in the acceptability of service history for certification credit (in the lead in paragraph of section 12.3.5), it is not referenced in the determination of error rates. For example, error rates can be computed for every version of the software product or it can be computed for all versions combined. This relates to the problem of what are the rules for computing the time duration, i.e., the rules for resetting the clock. This also brings up the point of whether the applicant can choose to use only the last 2 years of contiguous service history during which the software may be more stable compared to the very first fielding of the product. The guidance is silent on how long is long enough for a particular level, whether the length of service history is important, definition of error rates, definition of stability, and how to use error rates in system safety assessment to arrive at what rates are acceptable for a given criticality level.	1. What is the actual error rate computed using the service history? 2. Is this error rate greater than the proposed acceptable error rate defined in PSAC, according to J(4)? 3. If the error rate is greater, was analysis conducted to reassess the error rates?	Question of Problem Reporting

2.3 RELATIONSHIP WITH PREVIOUSLY DEVELOPED SOFTWARE.

DO-178B uses the term previously developed software (PDS) to describe software that falls in one of three categories:

- Commercial Off-The-Shelf (COTS) software,
- Software developed to a standard other than DO-178B, and
- Software developed prior to DO-178B.

By this definition, it is hard to imagine an instance when a product service history argument will be made on software other than PDS. DO-178B provides specific guidance for PDS that should be used in conjunction with the contents of this report when seeking certification credit. Combining alternative methods to meeting one or more objectives is best accomplished by conducting a gap analysis designed to determine where data may be insufficient to clearly demonstrate compliance with the objective. Such an approach is described in DO-248, discussion paper No. 5.

2.4 PRODUCT SERVICE HISTORY VERSUS SOFTWARE RELIABILITY.

The DO-178B definition of product service history is very similar to the IEEE definition of reliability, which is "the ability of a product to perform a required function under stated conditions for a stated period of time." It should also be noted that DO-178B states the following paragraphs regarding software reliability:

> Section 2.2.3: *"Development of software to a software level does not imply the assignment of a failure rate for the software. Thus, software levels or software reliability rates based on software levels cannot be used by the system safety assessment process as can hardware failure rates."*
>
> Section 12.3.4: *"During the preparation of this document, methods for estimating the post-verification probabilities of software errors were examined. The goal was to develop numerical requirements for such probabilities for software in computer-based airborne systems or equipment. The conclusion reached, however, was that currently available methods do not provide results in which confidence can be placed to the level required for this purpose. Hence, this document does not provide guidance for software error rates. If the applicant proposes to use software reliability models for certification credit, rationale for the model should be included in the Plan for Software Aspects of Certification, and agreed with by the certification authority."*

The effect of these two statements has been a virtual moratorium on the application or even exploration of software reliability as an alternative method for satisfying DO-178B.

This creates an inherent difficulty for the product service history approach as well, since service history arguments are largely predicated on the residual error rates or the probability of latent software errors remaining after verification. The authors of DO-178B side-stepped this issue by allowing certification credit for service history based on qualitative assessments of the sufficiency and relevancy of the product service history.

3. THE ELEMENTS OF PRODUCT SERVICE HISTORY.

As noted in the previous section, the topic of product service history may be examined by looking at the various elements that comprise its definition. For the purposes of this report, four components were defined: problem reporting, operations, environment, and time. Considering each of these components separately results in different but interrelated sets of questions that must be asked when the use of product service history is being considered. The questions have been broken into these classes only to simplify the problem. Answers to these questions must satisfy both the relevancy and sufficiency criteria discussed in Section 12.3.5 of DO-178B.

This section provides a discussion of each set of questions arising from the product service history definition. One representation of these questions is provided in the form of worksheets. While these worksheets may be adapted or tailored to fit a particular project, users are cautioned to maintain an objective view when evaluating service history data. As illustrated in the sections below, even subtle changes in any one of the four areas can lead to unpredictable results when software is used in a new system or in a way not originally envisioned.

3.1 QUESTIONS OF PROBLEM REPORTING.

Questions of problem reporting are primarily the same as ones faced in configuration control and management. All of the elements of DO-178B Section 11.4, Software Configuration Management (SCM) Plan, apply. The problems have to be uniquely identified, they should be traceable to the version of software/product, a method of closing the problems must be defined, and closure of the problems must be accomplished with proper change control activity. Changes must be reviewed for priority of problems and change impact. Data on problems, corrections, and baselines must be kept under control to assure the integrity of data.

All of these activities are a natural part of a well-defined process. However, in the case of previously developed software (PDS), it is assumed that these activities are not visible to the user of the product. The vendor who is in charge of problem collection may not have a robust process. The vendor may not have a robust policy for classifying and prioritizing the problems. This is further exacerbated by multiple users employing the software in ways to which the vendor has no visibility.

When patches are installed, some users may install the patch, whereas many others may not. This means that some users are using the uncorrected version and some are using the corrected version. This results in a service history that cannot be treated as a monolithic block, rather, it must be distributed across the different versions. Only those versions with a clear similarity to the intended use may be used to arrive at the total product service history. There are numerous reasons affecting problem report classification and accuracy including:

- Not all users may be using the product per the user manual.
- Vendors may not have a complete, consistent, or accurate way of identifying those problems attributable to software.
- Not all users may be reporting all of the problems.

- Users may find work-around procedures and thus stop reporting all occurrences.

- Vendors may not require subsequent occurrences of a problem to be reported.

- Vendors may treat problems found internally differently from those found by their customers, thus underreporting the total number of problems experienced.

Problems may also be introduced while fixing other problems. Such problems should also be logged once the product is fielded. If some of the problems are unique to a particular small sector of users, the vendor may not fix the problem or may selectively provide a patch. Attention must be paid to the number and type of open problems. A vendor's policy for choosing which errors are to be fixed should also be noted in the qualitative assessment. A vendor may place priority on non-safety-critical problems reported by a large sector of users over safety-critical problems reported by a small sector of users.

Assignment of version numbers and tracking the operating versions of the product to be traced to the problems is a difficult task. If vendors provide patches for their software or frequently introduce revisions to the field, this must be taken into account in arriving at the total number of versions for which service history is valid, and for which the total service periods can be combined.

Visibility into how problems were fixed may be of use when the solutions affect the usage of the product in a safety-critical application (whether requirements were compromised, new assumptions were made, new requirements were added, new design features were added, change-impact analysis was conducted, list of affected requirements/assumptions are provided to user, any effect on hardware is noted, etc.).

Some vendors may be following certain regulations or policy regarding configuration control of the problem reporting process. Such policies may help in determining if service history data is clean. Some problems may also be corrected in periodic upgrades of the product. It is important to understand the vendor's policy for dissemination of patches, warnings, work-arounds, and upgrades. After a new version is disseminated, spikes in error rates need to be traced to assess the complexity of changes, the quality of change-impact analysis, the quality of the vendor's verification process, and the diversity of the product usage.

3.1.1 Individual Questions to Consider.

The key questions to be addressed in the area of problem reporting and configuration management for the purpose of establishing the minimum objective criteria for using service history include a good consistent definition of problems, classification of problems, tracking with respect to software versions, and tracking with respect to solutions.

Table 2 provides a set of questions in the form of a worksheet that may be used to evaluate the relevance and sufficiency of problem report/configuration management using the data available from product service history. Further, this table indicates where there are gaps in DO-178B

guidance with respect to problem reports and analyses that should be performed for calculation of error rates.

The following considerations, based on Section 12.3.5 of DO-178B, were used in formulating the questions in table 2:

- Data available on problems
- Data derivable from the problem reports
- Analysis to be performed
- Indications of supplemental verification

TABLE 2. WORKSHEET FOR QUESTIONS OF PROBLEM REPORTING

Area:	Problem Reporting/ Configuration Management	Software Being Evaluated:	
Project:		Evaluator:	
Date:			

	DO-178B Reference	Question	Response	Issues
1.	12.3.5 a and c	Are the software versions tracked during the service history duration?		
2.	12.3.5 a and c	Are problem reports tracked with respect to particular versions of software?		
3.	12.3.5 a	Are problem reports associated with the solutions/patches and an analysis of change impact?		
4.	12.3.5 a	Is revision/change history maintained for different versions of the software?		
5.	12.3.5 a	Have change impact analyses been performed for changes?		
6.	12.3.5 b	Were in-service problems reported?		
7.	12.3.5 b	Were all reported problems recorded?		
8.	12.3.5 b	Were these problem reports stored in a repository from which they can be retrieved?		
9.	12.3.5 b	Were in-service problems thoroughly analyzed and/or those analyses included or appropriately referenced in the problem reports?		
10.		Are problems within the problem report repository classified?		

TABLE 2. WORKSHEET FOR QUESTIONS OF PROBLEM REPORTING (Continued)

	DO-178B Reference	Question	Response	Issues
11.		If the same type of problem was reported multiple times, were there multiple entries or a single entry for a specific problem?		
12.		If problems were found in the lab in executing copies of operational versions of software during the service history period, were these problems included in the problem reporting system?		
13.	12.3.5 c	Is each problem report tracked with the status of whether it is fixed or open?		
14.	12.3.5 c	If the problem was fixed, is there a record of how the problem was fixed (in requirements, design, code)?		
15.	12.3.5 c	Is there a record of a new version of software with new release after the problem was fixed?		
16.	12.3.5 c	Are there problems with no corresponding record of change in software version?		
17.	12.3.5 c	Does the change history show that the software is currently stable and mature?		
18.		Does the product have the property of exhibiting the error with a message to the user? (Some products may not have error trapping facilities, so they may just continue executing with wrong results and with no indication of failure.)		
19.		Has the vendor (or the problem report collecting agency) made it clear to all users that problems are being collected and corrected?		
20.	12.3.5 h	Are all problems within the problem report repository classified?		
21.	12.3.5 h	Are safety-related problems identified as such? Can safety-related problems be retrieved?		
22.	12.3.5 h	Is there a record of which safety problems are fixed and which problems are left open?		

TABLE 2. WORKSHEET FOR QUESTIONS OF PROBLEM REPORTING (Continued)

	DO-178B Reference	Question	Response	Issues
23.	12.3.5.h	Is there enough data after the last fix of safety-related problems to assess that the problem has been corrected and no new safety-related problems have surfaced?		
24.	12.3.5h	Do open problem reports have any safety impact?		
25.		Is there enough data after the last fix of safety-related problems to assess that the problem is solved and no new safety-related problems have surfaced?		
26.	12.3.5 i	Are the problem reports and their solutions classified to indicate how a fix was implemented?		
27.	12.3.5 i	Is it possible to trace particular patches with release versions and infer from design and code fixes that the new versions correspond to these fixes?		
28.	12.3.5 i	Is it possible to separate the problem reports that were fixed in the hardware or change of requirements?		
29.		Are problem reports associated with the solutions/patches and an analysis of change?		
30.		If the solutions indicated a change in the hardware or mode of usage or requirements, is there an analysis of whether these changes invalidate the service history data before that change?		
31.		Is there a fix to a problem with changes to software but with no record of the change in the software version?		
32.	12.3.5 j(2)	Is the service period defined appropriate to the nature of software in question?		
33.	12.3.5 j(2)	How many copies of the software are in use and being tracked for problems?		
34.		How many of these applications can be considered to be similar in operation and environment?		
35.	12.3.5 j(2)	Are the input/output domains the same between the service duration and the proposed usage?		

TABLE 2. WORKSHEET FOR QUESTIONS OF PROBLEM REPORTING (Continued)

	DO-178B Reference	Question	Response	Issues
36.		If the input/output domains are different, can they be amended using glue code?		
37.	12.3.5 j(2)	Does the service period include normal and abnormal operating conditions?		
38.		Is there a record of the total number of service calls received during the period?		
39.		Were warnings and service interruptions a part of this problem-reporting system?		
40.		Were warnings analyzed to assure that they were or were not problems?		
41.	12.3.5 j(3)	Was there a procedure used to log the problem reports as errors?		
42.	12.3.5 j(3)	What was the reasoning behind the contents of the procedure?		
43.	12.3.5 j(3)	Is there evidence that this procedure was enforced and used consistently throughout the service history period?		
44.		Does the history of warranty claims made on the product match with the kind of problems seen in the service history?		
45.		Have problem reports identified as a nonsafety problem in the original domain been reviewed to determine if they are safety related in the target domain?		

3.1.2 Identified Gaps and Gap Closure—Problem Reporting.

There are considerable gaps in DO-178B's treatment of problem reporting and configuration management as it relates to product service history.

- Guidance on Error Detection/Reporting Methods

  The product may not have the property of exhibiting an error with a message to the user or other external symptoms. Some products may not have error-trapping facilities, so they may just continue executing with wrong results with no indication of failure. Such systems will have very few errors logged as problem reports. The gap that needs to be filled here is to include questions on whether service interruptions were a part of this problem-reporting system. In addition, warnings (not necessarily error messages) need to be analyzed to assess if they were truly problems.

- Guidance on Complete Reporting/Problem Report Format

  It may not make good business sense to a vendor to publish a large number of problems. One cannot assume that all of the problems that were reported were logged into the problem-reporting system. One way to check if any of the problems were left out from the logs is to check if there is a record of a total number of service calls received during the period and check this number against the log entries. Also, logs should be inspected to understand what separates true software problems from user errors. If software errors are classified as system features and user errors, the number of problems will be lower.

  There is no guideline in DO-178B for what constitutes an acceptable problem report. If the vendor had an acceptable policy on how to log problems as errors, logs should be examined to see if there is evidence that this policy was used consistently throughout the service history period. History of warranty claims made on the product should match with the kind of problems seen in the service history.

- Guidance on Consistency in Assessing Safety Effects

  If there are problem reports that are not fixed, the applicant needs to decide if there are safety-critical problems that are not fixed. To analyze the problem, one needs to understand the vendor's policy on resolving problems. The concern is that if the problem is found by a minority group of users, the vendor may not fix the problem—even if it is a safety problem. Classification of problems may or may not show these errors as critical problems, depending upon the classification of problems.

- Guidance on Correlation of Problem Reports to Specific Fixes

  There is no guidance on how problem reports, the corresponding fixes, and the version of the software related to one another. If the solutions indicated a change in the hardware or mode of usage or requirements, problem reports should be associated with the solutions/patches and an analysis of change. There should be an analysis of whether these changes invalidate the service history data before that change. If there was a fix to a problem with changes to software but with no corresponding record of change in the software version, this indicates that the software was not under proper configuration control. If there is no log entry within the problem-reporting system, correlation of problem reports to software errors may not be possible. If the problems are not categorized, it may be impossible to tell if safety problems are fixed. Imposing an acceptable list of minimum data items that must exist in the vendor's logs kept under configuration control can fill this gap.

- Guidance on Consistent Policy for Documentation of Problem Reports

  The analysis needs to consider the policy used for entry of problem reports. If the same type of problem was reported multiple times, there may be multiple entries or a single entry for a specific problem. A single entry for multiple occurrences of the problem will make the error rate smaller than it really is. On the other hand, if a user is impatient and

makes several calls for the same problem, these multiple calls may or may not be distinguished in the problem logs. If problems were found in the lab in executing copies of operational versions of software during the service history period, these problems may or may not have been included in the problem-reporting system, again skewing the error rate computation. Further, no consistent policy may have been applied; each technical support staff that took the report may have applied his/her own judgment on whether to enter the problem, as well as how to classify them. A common guideline on how to deal with problems and their classification is needed.

- Guidance on an Open Invitation to the Users to Report Problems

  Users may report the problems only when there is such an option available to them. It is important to know if the vendor (or the problem report collecting agency) made it clear to all users that problems are being collected and corrected. If the users were discouraged from reporting problems, there may not be as many service calls and that would make the error rates seem low. A request to see the published technical support policy may be required to trust that such an offer was made to users.

- Guidance on Objective Evidence of Adequate Repair Process

  DO-178B does not mention the need for objective evidence for an adequate repair process during the product history. Questions must be included in this regard to analyze problem data after the last fix of safety-related problems to assure that the problem is solved and that no new safety-related problems have surfaced.

- Guidance on Correlation of Problem Reports to Specific Copies Used in Specific Applications

  If there are many users using the products in many different applications, DO-178B requires an analysis of these applications for similar operation and environment. However, the guidance regarding this scenario is unclear. Certain questions need to be asked to bring forth additional information. When some of the applications in use are not considered to be of similar operation and environment, it is important to separate problem reports from those applications from all other problem reports. DO-178B is silent on safety problems reported in these applications. There may also be patches that are implemented into all copies because of problems in these nonrelevant applications. Guidance should be added that provides a correlation between operation, environment, and the problem report when multiple copies are in use.

- Guidance on Glue Code

  If the input/output domains are different, they can be, and often are, amended using glue code. DO-178B service history guidance stays silent on this possibility. It should be noted that glue code could be used to make the environment similar. Guidance should be added to consider glue code as developmental code.

- Guidance on Evaluating Problems in the Target Environment

  Problems that were previously analyzed and determined to be nonsafety related should be re-evaluated to determine if they might have a safety effect in the proposed (target) domain. This issue is not discussed in DO-178B. New guidance should require such an analysis.

3.2 QUESTIONS OF OPERATION.

The concept of operation is to define the usage characteristics of the software within the previous domain as compared with the target domain. These characteristics include people and procedures and the modes in which the service history was documented against the same items within the target domain. Figure 3 illustrates the type of comparisons that are needed.

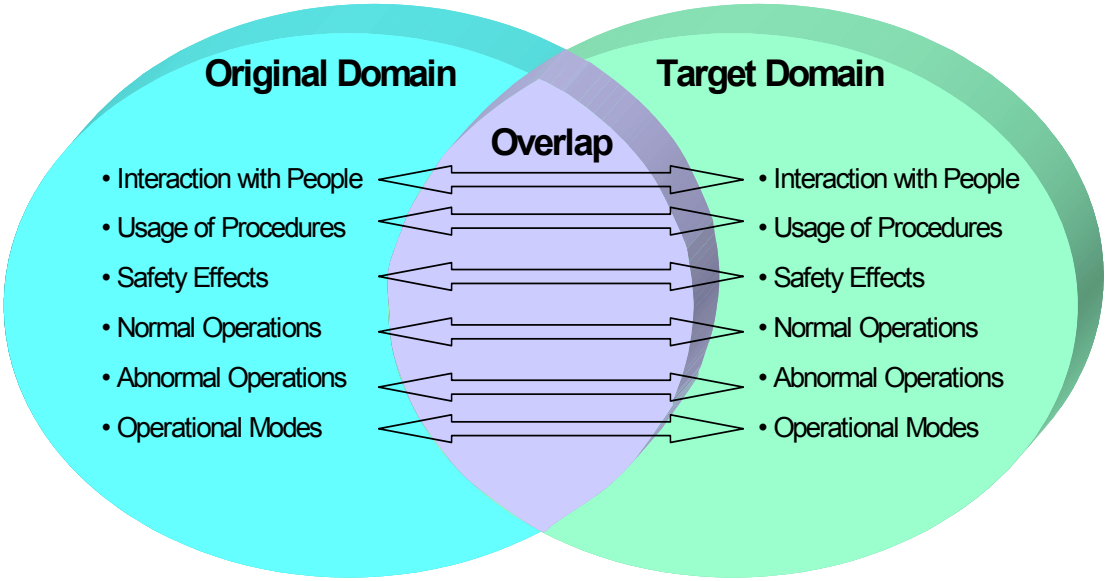

FIGURE 3. OPERATION

There are many concerns in evaluating similarity-of-operation that may not be so obvious. Where people and procedures are concerned, the training and qualifications of the people in the service history domain have to be noted so that the proposed (target) domain of usage can account for this by requiring similar operator training and qualification requirements.

Similarity in operational modes and the subset of software functions between the service history domain and the target domain are the main focus in this area. In general, it is expected that the functions to be employed in the target domain are a subset of those from the service history domain. Input and output domains may be evaluated in normal and abnormal operations to assess the completeness of coverage of all the functions that are to be reused in the target domain. This is the most fundamental basis for getting credit for service history by assessing that the software has a tried and tested history of pertinent functions.

Consider the case of ARIANE 5.[2] The self-destruction of the launcher was caused by the failure to mitigate the environmental differences between ARIANE 5 and ARIANE 4. Software reused in ARIANE 5 from ARIANE 4 included a function for the alignment of the strap-down inertial platform to be operative for 50 seconds. This requirement was based on a particular operational sequence of launch that is no longer used. Software exception generated from this portion of the code caused a chain of events that eventually led to shutting off the backup processor, errors in the primary processor causing an angle of attack of more than 20 degrees, separation of booster on the main stage, and self-destruction of the launcher. The reused function should have been operative only before liftoff or there should have been a thorough analysis of abnormal operating modes, differences in flight operations, and nominal range and value of parameters. There should have been a discussion of software exceptions and differences in actions taken to resolve these exceptions. Questions of operation and environment (discussed next) are highly interrelated. In this example, a study of target operations could have found the fault just as easily as a study of the target environment. Note also that what was a normal operation for ARIANE 4 became an abnormal operation for ARIANE 5.

The total availability of service history data may be much longer than what is considered similar operation. For example, there may be a total of 10,000 copies of particular software in use in the public domain, out of which only 10 copies may be in use in domains similar to the proposed usage. This would have a direct bearing on the ability to calculate error rates. This is discussed in more detail in Section 3.4, Questions of Time, of this document.

Modifications to the product during the service history interval need to be studied to understand if these modifications were made for correcting errors in a dissimilar domain (for which service history credit is not being sought). The question here would be to note if change-impact analysis has been performed to assure that the functions that are of consequence in the service history data have not been adversely affected. This is quite possible if the changes have affected either the assumptions or requirements in this area.

If the service history collection occurred when the software was being used at a lower criticality than the intended usage in the target domain, caution should be exercised in taking credit. The types and severity of errors, as well as open problem reports must be examined to assure that the service history gives the proper level of assurance.

It must be noted that the service history duration should ideally include both normal and abnormal operations to cover features such as redundancy, backup, other fault-tolerance techniques, and corner conditions. An analysis should be conducted to find which features were not exercised in the service history, so that supplemental verification can be performed.

If the product was used with different data/parameters (for example adaptation data, external data sets, internal parameters) in the service environment, these differences should be examined for possible risks in the target environment.

[2] "ARIANE 5 Flight 501 Failure," Reported by the Inquiry Board, the Chairman of the Board, Professor J. L. Lions, Paris, 19 July 1996.

3.2.1 Individual Questions to Consider.

The key questions to be addressed in the area of operation for the purpose of establishing the minimum objective criteria for use of service history data include an analysis for establishing similarity of operation between the service history and the proposed application. Service history data that reflect dissimilar operations cannot be used for computing service history duration.

Table 3 provides a set of questions in the form of a worksheet that may be used to evaluate the similarity of operation using the data available from the product service history. Further, this table indicates where DO-178B guidance is lacking with respect to similarity analysis, as well as analyses when dissimilarities are found.

The following considerations, based on Section 12.3.5 of DO-178B, were used in formulating the questions in table 3:

- Data pertinent to operation
- Derivable data associated with operations
- Analysis to be performed
- Indications of supplemental verification

TABLE 3. WORKSHEET FOR QUESTIONS OF OPERATION

Area:	Operation	Software Being Evaluated:	
Project:		Evaluator:	
Date:			

	DO-178B Reference	Question	Response	Issues
1.	12.3.5 d	Is the intended software operation similar to the usage during the service history (i.e., its interface with the external world, people, and procedures)?		
2.	12.3.5 e	Have the differences between service history usage and proposed usage been analyzed?		
3.	12.3.5 e	Are there differences in the operating modes in the new usage?		
4.	12.3.5 g	Are only some of the functions of the proposed application used in service usage?		

TABLE 3. WORKSHEET FOR QUESTIONS OF OPERATION (Continued)

	DO-178B Reference	Question	Response	Issues
5.	12.3.5 j(1), g	Is there a gap analysis of functions that are needed in the proposed application but have not been used in the service duration?		
6.	12.3.5 j(2)	Is the definition of normal operation and normal operation time appropriate to the product?		
7.		Does the service period include normal and abnormal operating conditions?		
8.		Is there a technology difference in the usage of the product from service history duration (manual vs automatic, user intercept of errors, used within a network vs standalone, etc.)?		
9.		Was operator training on procedures required in the use of the product during the recorded service history time period?		
10.		Is there a plan to provide similar training in the new operation?		
11.		Will the software level for the new system be the same as it was in the old system?		

3.2.2 Identified Gaps and Gap Closure—Operation.

- Guidance on the Analysis of Abnormal Conditions Within Normal Operations

  The service history may be a result of the software in question that is embedded in a larger system. In the larger system, the input to the software in question may be filtered or sanitized to avoid abnormal behavior. It is difficult or impossible to draw conclusions that the operating environments are the same, particularly if there are no plans to use the same glue code, partitioning, or error trapping in the proposed computer environment. The service history may be collected in the normal operating conditions, but the data that causes abnormal conditions may not have reached the software in question. The proposed method for filling this gap is to provide additional guidance on

  a. Documentation of the exact circumstances of operating the software in question
  b. The use of the same or similar wrapper code, glue code, or error-trapping code

- Guidance on Analysis of Technology Differences in Operation

  Technology differences in the usage of the product from service history duration (for example manual vs automatic, user intercept of errors, used within a network vs stand alone, etc.) may affect the argument of similar operations. DO-178B does not explicitly call for an analysis of such differences in operation. In the service history duration there may have been a user intercept or a procedural go-around for a problem so the problem was not reported as a problem. Additional guidance is needed in this area.

- Guidance on Consideration of Operator Training

  Operator training for performing expected procedures is part of the safety net in the use of the system. One may be comparing lack of errors in an operation where the operator is highly trained with an operational profile where no training is planned. DO-178B is silent on this aspect of similarity of operation. A trained operator may have been the reason for a low-error rate. If the same product is used with an untrained operator, there may be many problems that were not originally expected. Further guidance should be developed to address this issue.

- Guidance on Analysis of Similarity of Operation

  Problem data may or may not have all of the data needed to make a decision of what functions were covered by the product service history. In order to assess similarity of operations, the functions that were used in the service history should be noted. However, there may not be data available to make this assessment. Problem data in the area of that function may be the only clue to indicate that the function was used. Functions that did not cause a problem may not have left any clues regarding their usage. As a corollary, absence of problems may not mean that the function was perfect and operated without problems. That function may not have been exercised by any of the users. An examination of other data sources that are needed to understand the operation during the service history duration is warranted. Perhaps a survey of past users may be conducted to assess the functions that were exercised. Very focused testing on code functionality never or infrequently exercised in nominal operation may be the answer to check those functions or abnormal conditions that may not have been exercised in the service history duration. Possible research in running code through automated toolsets to find keywords associated with abnormal processing or code structure is a suggested method for filling this gap. This is effectively supplementing service history approach with focused reverse engineering.

3.3 QUESTIONS OF ENVIRONMENT.

Questions of environment were separated from the questions of operation in order to distinguish the immediate computer environment in which the service history data was collected. This particular set of questions is tailored to address and mitigate software errors, which have their origin in hardware errors, interface, or resource assumptions. It should be noted that the exception handling and fault tolerance of the product whose service history is tracked should be

separated from the larger system in which it is embedded to assure the robustness of the product. This knowledge allows for an appropriate reuse of the product in the new environment. Figure 4 illustrates the items that should be considered in this area.

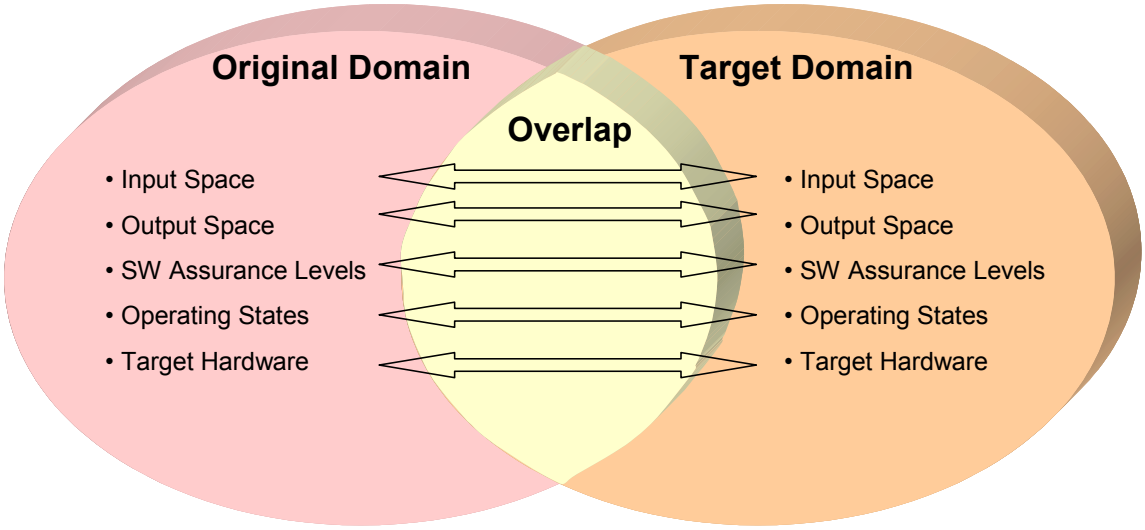

FIGURE 4. ENVIRONMENT

Similarity of environment may be assessed using the history of modifications to the product due to the particular hardware platform or because of resource requirements in the service environment, and the similar types of modifications needed to the product in the target environment.

Consider the example of the Patriot system failure to intercept the El Hussein (Scud) missile in Dharan.[3] Operational specifications for the system matched with the way the system behaved. However, the problems in the software were bugs only AFTER the operational environment had been redefined. The weapon was used, not in the detection and interception of aircraft, but rather, in the detection and interception of land-launched missiles. In its new capacity, the software failed because (1) there were missiles in a speed range that could and should be attacked and (2) the Patriot systems' primary mission would NOT be defending against hostile aircraft over a relatively short attack time, but rather, defending against a potential land-launched missile threat over many days. System performance degradation due to uncompensated clock drift crippled the weapon's defensive capability after the system had been continuously powered for days rather than the hours it was designed for. Unlike the ARIANE 5 case, it would have been difficult to detect the problems in this case since the system's failure was ultimately tied to the overall environment definition.

Service history credit should be counted strictly when the types of installations match the target environment, i.e., same or similar hardware platforms. Product literature may be reviewed to compare computer environments in terms of limitations and constraints such as resource usage.

[3] "Patriot Missle Defense Software Problems Led to Systems Failure at Dharan, Saudi Arabia," GAO Report February 1992, B-247094

If the problem reports identify problems because of usage in a particular computer environment different from the target environment, and the changes were made to fix these problems, the effect of these changes in the target environment should be considered.

If the product was used with different data/parameters (for example, adaptation data, external data sets, and internal parameters) in the service environment, these differences should be examined for possible risks in the target environment.

3.3.1 Individual Questions to Consider.

The key questions to be addressed in the area of environment include assessing the computing environment to assure that the environment in which the software was hosted during the service history is similar to the proposed environment. This analysis must include not just object code compatibility but time and memory utilization, accuracy, precision, communication services, built-in tests, fault tolerance, channels, ports, queuing models, priorities, and error recovery actions, etc.

Table 4 provides a set of questions in the form of a worksheet that may be used to evaluate the similarity of environment using the data available from the product service history.

The following considerations, based on Section 12.3.5 of DO-178B, were used in formulating the questions in table 4:

- Data pertinent to the computer environment
- Derivable data associated with the computer environment
- Analysis to be performed
- Indications of supplemental verification

TABLE 4. WORKSHEET FOR QUESTIONS OF ENVIRONMENT

Area:	Environment	Software Being Evaluated:	
Project:		Evaluator:	
Date:			

	DO-178B Reference	Question	Response	Issues
1.	12.3.5 e	Are the hardware environment of service history and the target environment similar?		
2.		Have the resource differences between the two computers been analyzed (time, memory, accuracy, precision, communication services, built-in tests, fault tolerance, channels and ports, queuing modes, priorities, error recovery actions, etc.)?		
3.		Are safety requirements encountered by the product the same in both environments?		
4.		Are exceptions encountered by the product the same in both environments?		
5.	12.3.5 f	Is the data needed to analyze similarity of environment available? (Such data are not usually a part of problem data.)		
6.		Does the analysis show which portions of the service history data are applicable to the proposed use?		
7.		How much service history credit can be assigned to the product, as opposed to the fault-tolerant properties of the computer environment in the service history duration?		
8.		Is the product compatible with the target computer without making modifications to the product software?		
9.	12.3.5 e and j(2)	If the hardware environments are different, have the differences been analyzed?		
10.		Were there hardware modifications during the service history period?		
11.		If there were hardware modifications, is it still appropriate to consider the service history duration before the modifications?		
12.		Are software requirements and design data needed to analyze whether the configuration control of any hardware changes noted in the service history are acceptable?		

3.3.2 Identified Gaps and Gap Closure—Environment.

- Guidance on Fault Tolerant Properties of the Service History Environment

  Software anomalies in the problem reports may be due to the lack of fault tolerance, error trapping, or built-in-testing in the computing environment in which the service history was collected. On the other hand, the fault tolerance properties of the computing environment may make the service history data look better than it should. The biggest gap identified in this area relates to the quality of data assumed in any service history argument. There is no guidance in DO-178B that directs the applicant to investigate whether software anomalies in the software under consideration for use in a new environment were being suppressed by the presence of other factors in the old installation. There have been techniques such as applying fault tree analyses to the various components to estimate the errors that are due to the software in question. Unfortunately, data to enable such an analysis is seldom available. Guidance should be developed for analyzing the amount of service history credit that can be assigned to the product, as opposed to the fault-tolerant properties of the computer environment in the service history duration.

- Guidance on Comparison of Computer Environments

  There is no guidance in DO-178B to direct the applicant on the types of analyses that must be conducted to understand computer environment differences. Resource differences such as time, memory, accuracy, precision, communication services, built-in tests, fault tolerance, channels and ports, queuing modes, priorities, and error recovery actions, etc., should be considered. Safety requirements and exceptions encountered by the product should be included in analyzing the similarity of environment. If software modifications have to be made compliant with the target hardware, analysis should be conducted to assess the quantity of service history data that is still valid. Guidance is needed in this area.

- Guidance on Consideration of Hardware Modifications During Service History Duration

  DO-178B does not give guidance for dealing with hardware modifications during the service history duration. All of the data needed for assessing if the same target hardware was used during the entire service history duration, may not be available. The data may have to be gleaned from other sources such as the history of the product. If there were changes to hardware during the service history duration, analysis should be conducted to assess if it is still appropriate to consider the service history duration before the modifications. Guidance is needed on hardware modifications during service history duration.

- Guidance on Consideration of Hardware Modifications for Computation of Error Rates

  DO-178B does not provide guidance to consider hardware modifications during the service history. If hardware was changed during the service history period, analysis should be conducted to assess if data before the modification can be used for computing

error rates. Analysis should also include considerations of whether software changes were made because of hardware modifications. New guidance is needed in this area.

3.4 QUESTIONS OF TIME.

There are many different ways of measuring the service history duration; duration may be measured without considering the changes made to the software or the clock may be restarted at corrections to safety problems. This question is related to how problems are classified and the vendor's priority system for correcting the problems. Figure 5 illustrates one common approach to measuring service history duration.

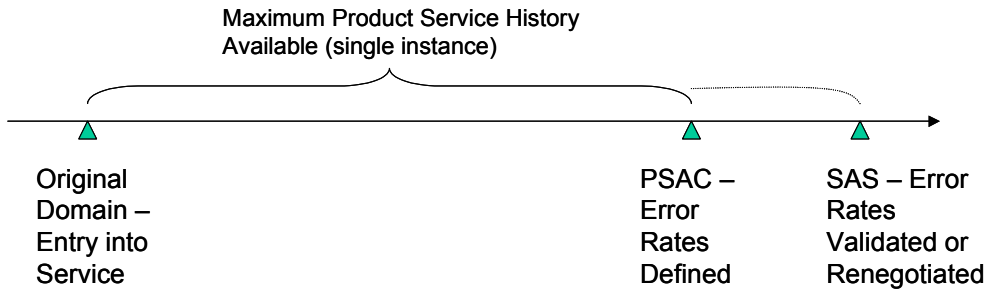

FIGURE 5. TIMELINE

The question of defining time relative to certification or continuing airworthiness has its parallels in other areas of the FAA. For example, following the Aloha Airlines incident in 1988, the National Transportation Safety Board noted, as part of their findings, that there appeared to be confusion in the terms flight cycle versus flight hour. The FAA released a Flight Standards Handbook Bulletin (HBAW 94-05B) to address this confusion as it related to aircraft maintenance.

The premise for using service history is based on the assumption that service history data gives us the evidence that all of the required functions have been repeatedly exercised and are correct. Strictly speaking, this assumption has no bearing on time at all. Time comes into the picture only because there is a comfort in a statistical sense that the probability of exercising all of the needed functions is greater as more time passes.

Time is further modified within the definition by the need for its measurement to take place over a contiguous period. This qualification is designed to eliminate the potential for periods of improper execution or dormancy to be suppressed, thus skewing any conclusions drawn about the software under consideration.

A close review of the DO-178B product service history guidance produces additional terms that directly relate to time, most notably "hours in-service" and "normal operation time." These sound suspiciously like terms used to arrive at reliability numbers for hardware. An applicant wishing to use product service history is asked to describe in their Plan for Software Aspects of Certification (PSAC) the rationale for choosing a particular number of hours in service, including how normal operation time is defined. Consider a software function that is only used during

landing. It hardly seems reasonable to define in-service hours as flight time when the landing phase, during which the software is being exercised, accounts for only a small fraction of this overall time.

While DO-178B is silent on whether the contiguous time period varies with software level, all of the discussions within SC-190, SC-180, and CAST have tended to accept this as an axiom. Likewise, the assumption is always made that what is being discussed, in some way, is measurable using hours, minutes, days, etc. Generally, it is felt that attempting to categorize software execution in terms of clock cycles, frames, or states is something for which sufficient data would be impossible to directly measure and would ultimately rely on inference from a clock-based measurement.

DO-178B in Sections 12.3.5 j(2) and k refer to computation of error rates. DO-178B does not provide specific guidance as to how this computation should be performed or what specific data is be used. This provides the applicant with a fair amount of flexibility in the application of the service history argument. Error rates could be defined as number of errors divided by the time duration. In some cases, time duration is not as relevant as using the number of events such as takeoffs or landing, flight hours, flight distance, total population operating time, or only the number of times an operator queried the software for specific information. For use in this computation, the duration should be analyzed to be relevant. DO-178B does not specify whether all errors are considered to be of the same weight in these computations. It seems logical that even a few safety errors should be of higher consequence than a large number of nonsafety errors. Although there is a discussion of safety problems in Section 12.3.5 h, there is no indication of how these problems are used in error-rate computations.

Note: Grounds for restarting the clock within the service history duration is not discussed in DO-178B. When there is a major software update or a hardware change, whether the data that is collected before the changes can be counted, as service history duration for measuring the error rates for the new software, is the question. In a number of software reliability models, time is reset when major changes are made since the software that was tracked is no longer the software that is used. There are other models that compensate for changes in software. This gap is tied to whether software reliability models can be used, and if so how do you assess that correct assumptions are made in the use of a particular model in a particular circumstance. This is illustrated in figure 6.

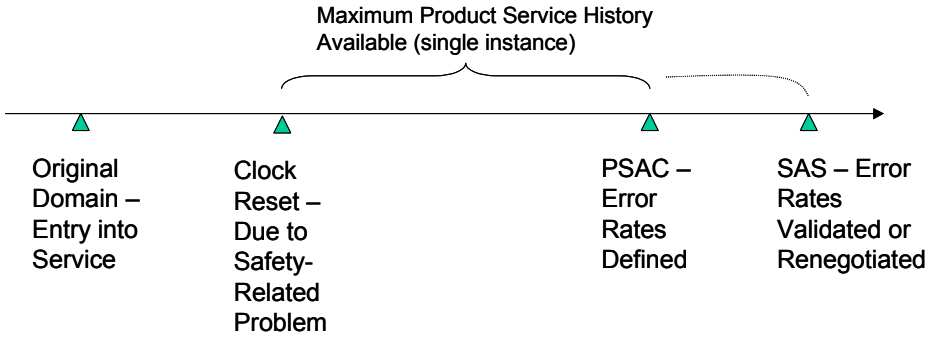

FIGURE 6. CALCULATION OF SERVICE HISTORY DURATION

3.4.1 Individual Questions to Consider.

The key questions to be addressed in the area of time for the purpose of establishing the minimum objective criteria concerning service history include units of measurement in the service history duration definition as appropriate to the usage of the software in question, reliability and consistency of time measurement, and justification for duration used in the calculation of error rates.

Table 5 provides a set of questions in the form of a worksheet that may be used to evaluate service history time duration and error rates using the data available from product service history.

The following considerations, based on Section 12.3.5 of DO-178B, were used from formulating the questions in table 5.

- Pertinent data related to time
- Derivable data regarding time
- Error rate considerations
- Analysis to be performed
- Indications of supplemental verification

TABLE 5. WORKSHEET FOR QUESTIONS OF TIME

Area:	Time	Software Being Evaluated:	
Project:		Evaluator:	
Date:			

	DO-178B Reference	Question	Response	Issues
1.	12.3.5 j(2)	What is the definition of service period?		
2.	12.3.5 j(2)	Is the service period defined appropriate to the nature of software in question?		
3.	12.3.5 j(2)	What is the definition of normal operation time?		
4.	12.3.5 j(2)	Does normal operation time used in service period include normal and abnormal operating conditions?		
5.	Glossary	Can contiguous operation time be derived from service history data?		
6.		Is “applicable-service” portion recognized from the total service history data availability?		
7.	12.3.5 j(2)	What was the criterion for evaluating the service period duration?		

TABLE 5. WORKSHEET FOR QUESTIONS OF TIME (Continued)

	DO-178B Reference	Question	Response	Issues
8.	12.3.5 j(2)	How many copies of the software are in use and being tracked for problems?		
9.		What is the duration of applicable service?		
10.		Is applicable-service definition appropriate?		
11.		Is this the duration used for calculation of error rates?		
12.		How reliable was the means of measuring time?		
13.		How consistent was the means of measuring time throughout the service history duration?		
14.	12.3.5 j(4)	Do you have a proposed accepted error rate justifiable and appropriate for the level of safety of proposed usage, before analyzing the service history data?		
15.	12.3.5 j(4)	How do you propose that this error rate be calculated, before analyzing the service history data?		
16.		Is the error rate computation (total errors divided by time duration, by number of execution cycles, by number of events such as landing, by flight hours, by flight distance, or by total population operating time) appropriate to the application in question?		
		What was the total duration of time used for this computation? Has care been taken to consider only the appropriate durations?		
17.	12.3.5 k	What is the actual error rate computed after analyzing the service history data?		
18.	12.3.5 k	Is this error rate greater than the proposed acceptable error rate defined in PSAC, according to j(4)?		
19.	12.3.5 k	If the error rate is greater, was analysis conducted to reassess the error rates?		

3.4.2 Identified Gaps and Gap Closure—Time.

- Guidance on Minimum Criteria for Evaluation of Configuration Management and Quality Assurance of Problem Reports

  DO-178B is silent on minimum criteria for configuration management and quality assurance of problem reports for evaluating a service period duration. Guidance is needed in this area.

- Guidance on Acceptable Service History Duration

  Definition of acceptable service duration is missing in DO-178B. Current guidance states that the error rates have to be predicted at the time of planning. Error rates should be computed using the service history period. There are arguments that the acceptable period does not have to be in time, since what is important here is the extent of usage. There is also an argument that the users usually underreport the problems. If the problem reports are tracked for a reasonable length of time, there is a chance that problems, if any, are reported which may be one reason to collect data for a long period of time. Thresholds for service history duration may vary by software criticality level. Thresholds cannot objectively vary with time unless a connection can be made with the safety requirements of that level. SC-190 discussions for CNS/ATM software did indicate "engineering judgment" as a subjective measure for the length of time. Thresholds may vary by code complexity or similar architectural measure (code size, degree of coupling, etc.). Such measures may not be practical, since there is no guarantee that the software structure is known. However, guidance is needed in this area.

- Guidance on Definition of Normal Operation Time

  Definition of normal operation time is left open in DO-178B. It is presumed that normal operation time is the time of operation similar to expected operation in the proposed environment. These conditions of operation should include similar external stimuli as input into the software; the number of data and control inputs, the input characteristics, the frequency of the input, accuracy and precision of input, the statistical spread of input, the combinations and permutations of input, etc., should be the same as operations in the proposed environment. Guidance on such considerations is needed.

- Guidance on Restarting the Clock Within Service History Duration

  Grounds for restarting the clock within the service history duration is not discussed in DO-178B. When there is a major software update or a hardware change, whether the data that is collected before the changes can be counted, as service history duration for measuring the error rates for the new software, is the question. In a number of software reliability models, time is reset when major changes are made since the software that was tracked is no longer the software that is used. There are other models that compensate for changes in software. This gap is tied to whether software reliability models can be used, and if so, how do you assess that correct assumptions are being made pertaining to a particular model in a particular circumstance? Guidance is needed in this area also.

- Guidance on Methodology for Assessing Corner Condition Processing Time

  The total service history duration may not reflect how much of the time was spent in routine processing versus exception handling or corner condition processing. Research could be conducted in this area to derive a method to find average rates of processing time spent in nominal processing versus exception processing. Such a method would provide guidance to fill this gap.

- Guidance on Computing Error Rates Using Service History From Repeated Use of Software

  DO-178B is silent on how to combine multiple instances (see figure 7) of software used to derive total duration and total error rates. Attempting to do this may lead to cases where different error rates are observed. This would imply differences in the way errors are collected and logged. It may also imply that the operation and computer environments are not truly similar. Guidance is needed to address these issues.

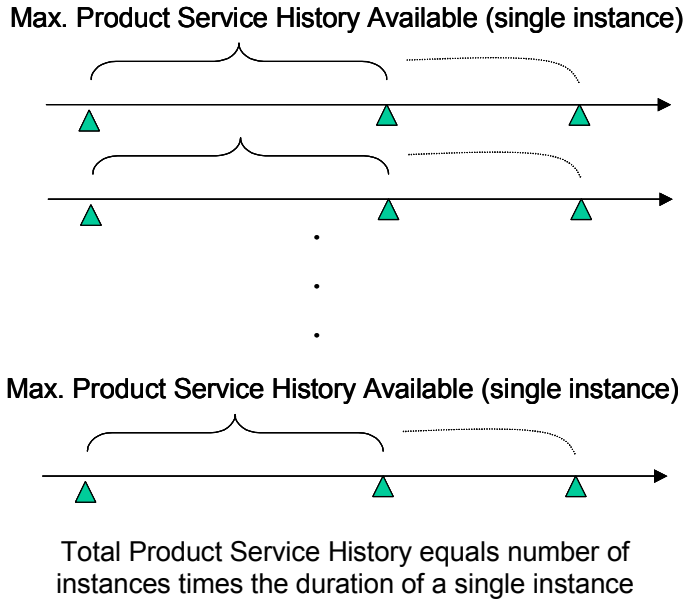

FIGURE 7. PRODUCT SERVICE HISTORY—MULTIPLE OPERATING COPIES

4. ADEQUACY OF DEVELOPMENT PROCESS.

DO-178B gives guidance for both process and product assurance. Product service history does not provide any direct objective evidence of the process used in creating the software for which it describes. Applicants wishing to make use of product service history must determine a way of demonstrating compliance with the objectives of DO-178B. This generally involves complementing product service history with additional alternate methods.

Numerous attempts have been made to equate specific objectives for which product service history could be "traded with." Such attempts within SC-190 and CAST actually involved the creation of tables listing the objectives of DO-178B and stating for each objective whether service history could fully or partially satisfy the objective. These attempts were reviewed as part of this research in hopes that something had been overlooked that prevented their acceptance by the broader community. The inherit problem is the unquantifiable nature of the processes used to create and assure software. DO-178B is based on the premise that a good development process yields a better product; one that is more likely to perform its intended function and less likely to perform unintended functions. By its very nature, this is a qualitative approach, rather than a quantitative one.

The 66 objectives of DO-178B are divided across six main areas: planning, development, verification, quality assurance, configuration management, and certification liaison. The definition of product service history really only addresses two of these. The first is fairly direct, namely, problem reporting and configuration management of the software the data describes. The second is verification of the code to some degree by virtue of its execution. In fact, a cogent argument can be made that service history represents at least some amount of a variety of testing techniques including:

- Stress testing
- Random testing
- Scenario-based testing
- Regression testing
- Accelerated life testing
- Exhaustive testing
- Domain testing
- Error guessing

All of these techniques may be used to accomplish one or more of the verification objectives outlined in DO-178B. Frequently, they are elements of blackbox testing in controlled environments (either laboratory or airplane) in typical DO-178B projects. The good news is that about 60% of the objectives in DO-178B are verification objectives. The bad news is that there would not seem to be any corollary to product service history for planning, development, quality assurance, and certification liaison during the original development of the software that the service history data describes.

With this in mind, it would seem most appropriate to focus specific attention on things that may be done to gain confidence in these other areas. If any development records are still available from the original vendor, these may be reviewed to gain confidence in the process that was followed. Such records could include the requirements documents, design data, quality assurance data, and test data that may supplement the service history data. In this last case, special attention should be paid to testing accomplished for error-handling routines, performance testing, and other testing focused on the robust characteristics of the software. Remember that these are the parts of the code least likely to have been exercised as part of the service history.

Confidence in the supplier's development process may also be gained through careful analysis of the problem report data collected over the service history period. In addition to the items discussed at the beginning of section 3, trend data may be analyzed to determine how well the supplier accomplishes reverification and whether the software does, in fact, appear to be maturing over time. This type of analysis is not directly discussed in the DO-178B, but is generally accepted by the software community.

Note that each of the above approaches can be stated in the negative as well. Spikes in problem reports right after a major build or a patch may indicate that the software is not maintainable or the quality of updates is not quite high. Recurring or chronic problems that go unresolved may also indicate poor processes.

5. ESTABLISHMENT OF "EQUIVALENT SAFETY".

Within a month and half of its publication, DO-178B was formally recognized via Advisory Circular (AC) 20-115B as a means, but not the sole means, for securing FAA approval of software in airborne systems. For new projects started after this AC was published, compliance with DO-178B had to be demonstrated. Those who have sought to use other approaches for securing FAA approval have generally been required to show how their approach met the intent behind the DO-178B objectives.

When discussing product service history, one of the most basic issues to understand is what the service history is demonstrating. Since the service history data generally exists for a system, typically a line-replaceable unit on an aircraft, any claim made for the software is an extrapolation from the system's performance. Systems are required to comply with one or more Federal Aviation Regulation (FAR), (more formally known as Title 14 Code of Federal Regulations (CFR)) before being certificated for use on an aircraft. A careful reading of DO-178B, along with the guidance governing certification of parts and equipment described in 14 CFR Part 21, shows that DO-178B is simply a means of satisfying the CFRs, specifically those elements describing intended function and absence of unintended function. The logical question that arises is whether service history can be used to demonstrate compliance directly with the CFRs.

While current guidance does not preclude such an argument, actually being able to demonstrate CFR compliance would be extremely difficult. Any attempt would need to overcome the basic issue of reliability applied to software. CFR XX.1309 uses terms such as extremely improbable to describe events that simply should not happen in the lifetime of a particular aircraft type. This has historically been translated to failure probabilities of 10^{-9} or better. There is no commercially accepted model written for software reliability that comes close to this number or that can be based on a realistic model.

Adding an unknown software component to a system is considered a safety risk. This safety risk has been under constant debate since software was first introduced to the flight management system in the early 1980s. For the time being, design assurance remains the only viable approach, with DO-178B serving as the most mature model for its application. There are, however, other ways of mitigating risk from an unknown software component. For example, architectural means may be employed to limit the effect of a software error leading to a system-level failure. Examples of architectural means include:

- Partitioning—preventing failures from noncritical functions affecting critical functions
- Wrappers—wrapper software prevents use of unneeded functionality, checks validity of parameters
- Software performance and safety monitors—credibility checks, redundant processors checking one another, fail-safe architectures
- External monitors—e.g., watchdog timers

Unfortunately, architectural means may not always be an option to protect against latent errors in software for which only service history data is available for certification credit. It may also be that the architectural means actually increases the system's complexity, thus potentially decreasing safety rather than increasing it. For higher criticality systems, service history may simply not be an appropriate or practical choice.

It is generally accepted that using service history data becomes easier when systems are relatively simple and are low criticality, the service history data is both abundant and of high quality, and the system's operating characteristics and external environment are close to the original systems.

6. RESEARCH SUMMARY.

The primary outputs of this research effort is a set of worksheets to facilitate consistent and complete assessments of service history data along with a detailed analysis of the impediments to applying service history arguments. An additional output is the identification of needed guidance, as well as follow-on research that may be useful in furthering the use of service history data while maintaining an equivalent level of safety. The remainder of this section discusses the results of this research in more detail with an emphasis on follow-on activities.

6.1 RESEARCH FINDINGS.

The preceding two sections served to summarize the current limitations in attempting to use service history to demonstrate compliance with the 66 objectives of DO-178B or even directly to the CFRs. As noted in section 3, gaps in the existing guidance can be found in every facet of the product service history definition. The lack of guidance addressing these issues has limited the use of service history as a viable alternate method. In cases where service history has been applied, these difficulties call into question the completeness and consistency of the arguments that were made and accepted.

Service history is data taken from real operations, unlike in simulated test environments where there may be mistakes in assumptions and data characteristics. Service history data is not systematic. Unlike a laboratory test setup, coverage of functions may not be known. If no errors were reported about a particular functionality, there is frequently no data to determine whether or not the functionality was not executed or if it was deemed to be error free. Resolving this ambiguity requires a coverage analysis for which no guidance exists. Assuming that such an analysis approach could be described and implemented, additional verification could be targeted at those elements of the software that were not executed. It is important to realize that if such an approach were viable, the duration of service history (i.e., how many times or how long the product was executed) would no longer matter, since instead of calculating error rates, the actual coverage of execution would be known.

It is a common misbelief that software with a sufficiently long service history duration can be used for safety-critical applications without further verification. Service history data is statistical in nature. There are many statistical models (software reliability) used for estimating the number and severity of software defects to provide management input for staffing, scheduling, quality, cost, and time to release. Any use of service history suffers from all of the same limitations these

current software reliability models suffer, most notably the assumption that errors are randomly distributed. Various approaches have been proposed to overcome this issue, most notably John Musa's contention that variability in the personnel creating the software, as well as variations in program execution relative to external factors do allow for the application of a random process model. This is an area for which additional study is needed before guidance can be formulated to truly allow service history use at higher criticality levels without additional process assurance.

If a statistical model were to be used in substantiating a claim using service history, validation of the model needs to be considered. One novel approach suggested during the breakout session of the FAA National Software Standardization Conference was the application of tool qualification for the model. Criteria would need to be developed on how such a qualification effort could be accomplished.

Another area in which further exploration may be warranted is in the development of "safety cases." A common approach to the evaluation and approval of safety-critical systems in Europe is the development of a comprehensive plan for meeting the required safety criteria, called a safety case. Specific approaches for building safety cases have been defined for IEC 61508. It may be possible to create a similar set of approaches for compliance with DO-178B and the related elements of the CFRs.

The development of an objective model for evaluating problem-reporting data is needed to facilitate using service history data. There is evidence that users in the field do not generally report all of the failures that they encounter. This observation works against using service history for credit since the data may be underreporting the failure and, hence, overreporting the reliability of software (and hardware). Some users may not take the time to report failures while others may not understand all of the system requirements (what the system is or is not supposed to do). There may also be some errors that are not clearly detected with error messages by the system. Likewise, errors attributable to the software may not be identified as such due to the system complexity or other masking factors. In calculating error rates, these kinds of errors must be compensated. This cannot be accomplished if they are not initially identified.

Some concerns (availability, integrity, reliability, maintainability, usability) are treated or weighted differently in other industrial sectors. Products from these domains may need to be evaluated using the assumptions from the initial industry that created or used them. Currently, no guidelines exist for this type of an evaluation. The FAA could publish guidelines on how to collect and document service history data. Industry now has the opportunity to glean this data from a number of data collection programs that are implemented through military and airline programs, and similar to telemetry data collection, during flight tests and shadow operations. Such guidelines could build on the question models contained herein, with the possible addition of weighting for each question based on that question's relative contribution to the integrity and usefulness of the data.

Other items that suggest further exploration include:

- Combining existing service history data with a period of shadow operations in a non-safety-critical environment (laboratory or test rack onboard an aircraft);

- Development for detailed criteria for evaluating problem reports for safety effects, both in the original domain and in the target domain;

- Definition of a similar concept to the Operational Services and Environment Definition (OSED) (defined in DO-264), to allow objective comparisons between two operations and computer environments; and

- Development of criteria for subtracting out the effects of system architecture and installation on service history data for the purposes of isolating the contribution of software to overall system error rates.

6.2 FOLLOW-ON PRIORITIZED RESEARCH NEEDS.

Recommendations for new guidance on product service history and for furthering the gap closure work includes (in order of importance):

a. Elimination of inconsistencies in DO-178B between using service history and prohibiting software reliability use in the assessment of system safety.

b. Development and publication of guidelines for using specific software reliability models, only if the models can be justified by means of tool qualification.

c. Provision of additional guidance to augment Section 12.3.5 of DO-178B as suggested by the gaps in question areas including the possible development of weights for each question area.

d. Creation of a requirement for additional design assurance activities when using service history at level A and B (consistent with DO-254).

e. Formulation of a tiered approach to service history duration at different levels based on either system safety (as computed from error rates) or on engineering judgment.

f. Publication of minimum requirements for the contents of problem reports (extension of DO-178B, Section 11.17 specifically for service history arguments).

g. Development and publication of guidance for making a safety case compliant to DO-178B.

Item a above is by far the largest and most difficult of these items. It also needs to be accomplished before or, at least, in parallel with item b. Item c covers a myriad of topics and has the potential for removing at least some of the current roadblocks to service history use. The development of a weighting system for the individual question discussed in both the report and handbook could be one way of adopting a risk-based approach similar to that discussed in AC 23.1309. Without a weighting structure, no framework exists by which to modulate the guidance given in DO-178B, Section 12.3.5, by software level. Item e represents another approach to achieving this, but is first dependent on items a and b being done and accepted by industry. Finally, item g represents a new paradigm within the aviation community. Although one could

argue that this was already accomplished, the mechanisms, language, and to some extent, the focus are different in an System Safety Assessment (SSA) from that found in a safety case.

7. CONCLUSION.

Service history is a powerful approach, which when used correctly, can make it possible to demonstrate the maturity of software that has previously been fielded and for which good data regarding its operation is available. To accomplish this, careful attention must be paid to a number of questions concerning the application of service history. Section 3 of this report (including the worksheets from appendix A of the Service History Handbook) discussed these questions in detail. Sections 4 and 5 helped to place those questions in the context of design assurance and safety, two fundamental aspects of creating and assuring software for airborne systems and equipment.

Service history, as an alternate method for demonstrating compliance to the objectives of DO-178B, is only one of many approaches that may be taken to demonstrate software maturity. When contemplating its use, one must be careful to consider the relationship between service history and software reliability. As noted in section 2, software reliability remains controversial, and cannot be used quantitatively in a safety assessment. Careful attention must be applied when defining error rates for a particular application and their definition should be discussed with the appropriate ACO at the onset of the certification project.

As more confidence is gained in the use of service history arguments for supporting certification efforts, the FAA may develop additional guidance. This report provides a first step in identifying where such guidance is needed, as well as the research that may be required to produce such guidance. It is also expected that this subject will be revisited when DO-178B is revised in the future. In the interim, it is hoped that this report, along with the handbook will help applicants apply service history arguments in a more thorough and consistent manner. Use of these documents by the FAA should allow for more consistent expectations being placed on applicants, something that has generally been shown to help control costs associated with achieving certification.

In conclusion, there is an inherent contradiction within DO-178B concerning service history as an alternate method and its disallowance of software reliability as it relates to system safety. If service history is to be a viable alternative method, significant work remains in the development of clear, concise, and scientifically valid guidance for its use. This report has provided an analysis of DO-178B content, gaps in that content, information concerning the topic from other industries, and a number of approaches for closing the gaps beginning with the inclusion of detailed worksheets for framing a service history evaluation.

APPENDIX A—DATA COLLECTION AND SYNTHESIS

Data collection and synthesis was accomplished by studying the research papers, by interviewing industry practitioners from different domains, and by holding an industry breakout session with a group of engineers from the aviation domain. All of the data that was collected was used in the gap analysis and gap closure plan. Much difficulty was observed in finding literature for service history usage on projects using commercial software, which are seeking the latest technology—inherently opposing ideas.

A.1 DATA COLLECTION AND SYNTHESIS 1: RESEARCH PAPERS.

A.1.1 DATA COLLECTION QUESTIONNAIRE.

1. Is the definition of product service history the same as in the aviation industry?

2. If yes, is it used in this reference for gaining assurance on the product reliability?

3. How is it used?

4. What is the method of assessment of quality and quantity of service history data?

5. Is this industry concerned with the process of error correction and other development methods?

6. Is this treatment of service history transferable to the aviation industry?

    a. With modifications
    b. In conjunction with other means

7. Are there problems in usage of service history as proposed in this reference?

8. If so, how can these problems be addressed?

9. Are there any details of "how to" on collection of problem reports, configuration control of problems and corrections, version control of released products, assessment of error data, statistical significance of amount of data, etc.?

10. What are the industry/regulator concerns in the proposed methods?

11. How does the material within this reference contribute to the handbook?

    a. Within the context of the proposed outline
    b. Outside the context of the outline- new items need to be added to the outline

A.1.2 DATA COLLECTION – LITERATURE SURVEY SELECTED NOTES.

U.K. Ministry of Defence, Defence Standard (9 December 1988) 00-40 (Part 6)/Issue 1 (ARMP-6) Reliability and Maintainability Part 6: In-Service R & M

Def Stan 0040 Part 6 is adapting ARMP-6 which is the NATO document "Allied Reliability and Maintainability Publication - 6"

The document describes a method of collecting data to assess the in-service reliability of systems. The purpose may be to assure the reliability or to improve the reliability. The same data may be used to augment certification data for a new system in which the product is being reused if it is in a similar environment. The questions at hand are how much data and what kind of data are needed for such purposes.

a. The data will state the type and numbers of systems/equipments for which service history is being collected.

b. The data will include the application and installation specifics as to whether it is installed in different ships, tanks, aircraft, and whether the application is in a standby or a fully operational mode.

c. All relevant information such as version numbers, scheduled maintenance actions, time duration of usage of the system (actual flight hours for aircraft), problem history, and corrective actions.

d. If the system is comprised of a group of smaller components, it is useful to track the history of individual systems.

e. It is prudent to record usage conditions such as climatic and mechanical conditions, the localized environment of the system/equipment, duty cycles, switching, electromagnetic environment, etc., as well as conditions such as storage and handling. Errors that may be due to these conditions can then be removed from skewing the statistics for the actual software errors. (Software errors can affect hardware and eventually cause system failures.)

f. The defect/failure data reporting, analysis, and categorization procedures should be clearly documented.

U.K. Ministry of Defence, Defence Standard (17 July 1992) 00-40 (Part 8)/Issue 1, Reliability and Maintainability Part 8: Procurement of Off-The-Shelf Equipment (ARMP-8)

Def Stan 0040 Part 8 is adapting ARMP-8 which is the NATO document "Allied Reliability and Maintainability Publication - 8."

The document describes considerations given to procurement of off-the-shelf (OTS) equipment including what to look for in the data supporting the reliability of the OTS equipment. The following points are noteworthy for service history data:

- New versions should be examined to determine if they were introduced to add new features or correct defects.

- Experience with faults that resulted in system shutdown or "crashes" should be of prime importance.

MoD 0055 Part 1 Issue 2, Requirements for Safety Related Software In Defence Equipment, Part 1: Requirements

The discussion of in-service history of previously developed software is less elaborate than DO-178B. Discussion here is purely qualitative and provided for merging into all other system documentation for completeness of considerations.

MoD 0055 Part2 Issue 2, Requirements for Safety Related Software in Defence Equipment, Part 2: Guidance

This has a relatively good discussion of in-service history usage. It requires both the software and the associated service history evidence to have been under configuration management throughout the software's service life.

Configuration changes during the software's service life should be identified and assessed in order to determine the stability and maturity of the software and to determine the applicability of the entire service history data to the particular version to be incorporated in the SRS (same as DO-178B).

The problem reporting system for the previously developed software should be such that there is confidence that the problems reported incorporate all software faults encountered by users, and that appropriate data is recorded from each problem report to enable judgments to be made about the severity and implications of the problems. (Notice that all software faults are to be recorded—this is stricter than DO-178B.)

This also advocates a safety case: The suitability of the quantified methods, assumptions, rationale, and factors relating to the applicability of the data should be justified in the Software Safety Case. The justification should include a comparison of actual error rates with the acceptability criteria defined in the Software Safety Plan and justification for the appropriateness of the data to the new application. (The comparison of error rates is similar to DO-178B.)

In cases where the previously developed software is justified on the basis of in-service history or extensive V&V and is treated as a black box, it may be acceptable for design information not to be provided as long as a comprehensive requirements specification for the software is provided (similar to DO-178B).

The operating environments of the previously developed software should be assessed to determine their relevance to its proposed use in the new application (similar to DO-178B).

This error rate computation follows similar guidelines as in DO-178B. But this standard goes further by asking for failure probabilities.

MoD 0056 Part 2 Issue 2 Safety Management Requirements for Defence Systems, Part 2: Guidance, 13 December 1996

Appendix D has guidance on using service history to allow validation of safety requirements by comparing safety requirements. "Arguments of similarity should not be used until any significant system problems experienced in service have been understood and resolved." The following points are made concerning service history:

a. Problem reporting procedures during the period of applicable service history were sufficient to provide an appropriate cross-section of in-service problems.

b. Changes to the referenced system or its operating instructions and environment during the service history period did not materially alter the safety of the system.

c. Proposed system modifications do not involve technology changes (for instance replacing a mechanical system with a software one).

d. Actual usage of the referenced system during the service history period was consistent with the intended usage of the new or modified system. If the operational environments of the existing and proposed applications differ, additional analyses should be conducted according to the requirements of this Standard.

Any reported safety related problems together with their causes and corrective actions should be analyzed in order to establish whether or not they are relevant to the proposed system, system modification or system application and the conclusions documented.

The other major point made in this document is the safety argument. "The Safety Case should contain well organized and reasoned justification clearly showing that:

a. Existing or supplemental analysis is sufficient to demonstrate that the system is tolerably safe.

b. Service history is applicable and changes to the referenced system configuration have been appropriately controlled and documented."

MoD 0042 Part 2 Issue 1, Reliability and Maintainability Assurance Guides, Part 2: Software, 1 September 1997

The document talks about an in-service software reliability case which is similar to the evidence that is being required in DO-178B. The argument is extended to particular methods, techniques, and processes rather than just software. Attributes to be used for similarity argument are:

- Data rates
- Throughput
- Functions used
- Input ranges
- Resource usage (e.g., CPU time)
- Configuration options (e.g., data conversion routines and thresholds)
- Operational mode
- Accuracy
- Mission time

Using field data in computing reliability is also discussed.

ISO/IEC 12207-1 Guide for Information Technology

This international standard is not intended for off-the-shelf software products unless incorporated into a deliverable product. This clause does not imply that the suppliers or developers of off-the-shelf software should not use ISO/IEC 12207 when developing, operating, or maintaining such software.

Additional guidance from ISO/IEC 12207 is to ensure that requirements are satisfied, documentation is available, and rights are satisfied.

The acquirer will consider options for acquisition against analysis of appropriate criteria to include risk, cost, and benefits for each option. Options include:

a. Purchase an off-the-shelf software product that satisfies the requirements.
b. Develop the software product or obtain the software service internally.
c. Develop the software product or obtain the software service through contract.
d. A combination of a, b, and c above.
e. Enhance an existing software product or service.

When an off-the-shelf software product is to be acquired, the acquirer will ensure the following conditions are satisfied:

a. The requirements for the software product are satisfied.
b. The documentation is available.
c. Proprietary, usage, ownership, warranty, and licensing rights are satisfied.
d. Future support for the software product is planned.

This reference promotes the use of off-the-shelf software product "as is." The full development process (5.3) may be excessive. Performance, documentation, proprietary matters, usage, ownership, warranty and licensing rights, and future support related to the software product should be evaluated.

Modification of off-the-shelf software product: Documentation may not be available. Depending on the criticality and expected future changes, the Development Process (5.3) should

be used via the Maintenance Process (clause 5.5). Performance, documentation, proprietary matters, ownership, usage, warranty and licensing rights, and future support related to the software product should be evaluated.

Software or firmware product embedded in or integral to a system: Since such a software product is a part of a larger system, the system-related activities in the Development Process (5.3) should be considered. In the system-related activities, only one verb "perform" or "support" needs to be selected. If the software or firmware product is not likely to be modified in the future, extent of documentation needs should be carefully examined.

Discussions of Software Service History in the Safety Critical news group, July 2000 (safety-critical@cs.york.ac.uk)

Discussion of COTS and COTS selection considerations on Carnegie-Mellon Software Engineering Institute web page http://www.sei.cmu.edu/cbs/papers/eval_bib.html

CBS Activity Areas have three primary categories of practices.

- Product and Technology evaluation
- Acquisition and Management
- Design and Software Engineering

Product and technology evaluation deals with user requirements and component qualification practices. The article supposes that it is not enough to have a system built of quality components; there are a number of other issues to consider when the system is to be used for an extended period of time. There is also a discussion of architectural mitigation of risks in using the product.

In the Acquisition and Management section, there is a discussion of business issues such as estimating costs, identifying and reducing risks, managing personnel and schedules, and flexible contracts. SEI has collected lessons learned from various projects.

Design and software engineering deals with glue code, software architecture, and working with the COTS manufacturers to come up with solutions to mismatched components.

There is no mention of service history in this article.

Other articles reviewed on this page are:

- "Evaluation of COTS Products: Some Thoughts on the Process," by David Carney.
- "Requirements and COTS-Based Systems: A Thorny Question Indeed," by David Carney.
- "COTS Product Evaluation and System Design," by David Carney.

- "COTS Evaluation in the Real World," by David Carney.
- "Simplex Architecture: Meeting the Challenges of Using COTS in High-Reliability System," by Lui Sha, John B. Goodenough, and Bill Pollak (through a link to Crosstalk-1998).
- "The Opportunities and Complexities of Applying Commercial Off-The-Shelf Components," by Lisa Brownsword, David Carney, and Tricia Oberndorf (through a link to Crosstalk – 1998).
- "Assembling Large Systems From COTS Components: Opportunities, Cautions, and Complexities," by David Carney.
- "Lessons Learned Applying Commercial Off-The-Shelf Products," by Lisa Brownsword.

There is no mention of service history in these articles.

Abts, Christopher, "COTS Software Integration Cost Modeling Study," Technical Report, University of Southern California, Center for Software Engineering, Barry Boehm, Director. The report (along with other information about COCOTS) is available at: http://sunset.usc.edu/COCOTS/cocots.html

This study was originally performed for the USAF Electronic Systems Center Hanscom AFB, Massachusetts. The study deals with only the economic feasibility of using a COTS product. This model does not deal with technical or strategic evaluation of COTS. There is just a mention of "mature products with a solid history of performance." No further discussion of the product history exists in either the paper or the web page.

Carney, David J. and Kurt C. Wallnau, "A Basis for Evaluation of Commercial Software," Information and Software Technology (1998) 851-860.

This article deals with the process, planning and individual responsibilities of the evaluation of commercial software for incorporation into systems. There is no discussion of product history.

Brown, A., and K. Wallnau, "A Framework for Evaluating Software Technology," IEEE Software, Vol. 13, No. 5 (Sep, 1996) pp. 39-49. Also "A Framework for Systematic Evaluation of Software Technologies," Component-Based Software Engineering, IEEE Computer Society, (Los Alamitos, 1996), pp. 27-40.
(http://www.sei.cmu.edu/cbs/papers/paper1.ps)

This article has an excellent idea of scoring the COTS using a set of criteria and weights assigned to them. The idea may be extended to apply to scoring findings within service history data. Criteria and weights for the use of particular COTS must be defined and product service history should be examined to derive the score.
Total score = Σ weight * individual score

Schneidewind, N.F., "Methods for Assessing COTS Reliability, Maintainability, and Availability," Proceedings of the International Conference on Software Maintenance (Bethesda, MD, Nov., 1998) IEEE Computer Society, Los Alamitos, pp. 224-225.

This paper deals with the environment in which COTS component under assessment is examined. It makes the point that the previous performance of a COTS component, as a part of a larger system, should be examined to extract the reliability estimates of the COTS component apart from the whole system in which it is embedded. The paper also makes the point that COTS should not be treated any differently from a custom product in the case of safety- (and mission) critical applications.

Graves, Karr, Marron, and Sly, "*Predicting Fault Incidence Using Software Change History,"* in IEEE Transactions on Software Engineering, Volume 26, Number 7, July 2000, pp. 653-661

This is a general paper on software, with many of the ideas applicable to COTS. This paper describes the aging process for software. Using change history, predictions of aging, and decaying of a software product may be made to indicate the current reliability of the product. Both product and process measures are used. Process measures used here are conclusions drawn from defect history data, namely, the number of changes, average age of the code, contribution of each change to the fault potential. Statistical models have been proposed to evaluate which characteristics of a module history were likely to indicate that it would see a large amount of faults. This paper is very useful in assessing COTS reliability using its service history.

Musa, J., Software Reliability Engineering, McGraw-Hill, 1998

This is a good introductory text on software reliability. The review was focused on possible application/relevance to failure reports from fielded software.

A relevant point that is discussed in this text is how to deal with unreported failures. Musa reports that it is observed that users in the field do not generally report all of the failures that they encounter. This observation works against using of service history for credit since the data may be underreporting the failure and, hence, overreporting the reliability of software (and hardware). He suggests compensating for the underreporting by observing the software update logs and nature of an error that must have occurred many times over before it was formally reported. Some users may not take the time to report failures while others may not understand all of the system requirements (what the system is or is not supposed to do). There may also be some errors, which are not as apparent, to be reported. Musa suggests models to compensate for unreported failures.

There is also an interesting discussion on failures of programs that have been fielded with no features added or faults removed (Poisson distribution) as opposed to failures of programs that are updated (step function).

Herrmann, D. S., Software Safety and Reliability, IEEE Computer Society, 1999

This book deals with basic definitions of safety and reliability and surveys various industries such as transportation, aerospace, defense, nuclear power, and biomedical, to assure safety and reliability. She also deals with non-sector-specific standards.

Within many of these discussions, such as NATO COTS Software Acquisition Guidelines and Policy Issues, there are points to be noted that are relevant to software service history. Upon studying this text, the following considerations that are specific to COTS were concluded:

1. How does the service history data of a component within a larger system translate to an indication of safety or reliability of that component which will be used as part of a different system?

2. Assessment of safety and reliability has to involve both qualitative and quantitative means.

3. Some concerns are unique to certain industrial sectors; COTS from other industry sectors should not be blindly applied to avionics systems.

4. Idea of product metrics from ANSI/IEEE STD 982.1 1989 and 982.2 1989 (measures to produce reliable software) can be extended to COTS products. Service history may be used to generate error, fault, and failure metrics.

Wood Alan, "Software Reliability Growth Models: Assumptions vs. Reality," Proceedings of the 8th International Symposium on Software Reliability Engineering (ISSRE "97)

The article is written for software in general, not just COTS. Whether or not a formal reliability model is used to assess COTS on the basis of the product service history, certain basic assumptions are questioned in this paper. Among these assumptions is the thinking that the software (SW) contains a fixed number of errors and with the passage of time more and more of these errors are found. It should be noted that while fixing errors, other errors might be introduced. Further, the defect finding techniques are to be questioned to assure that poor testing or insufficient usage do not make a bad COTS look good. A very good table of assumptions, reality, and effect on conclusions is presented.

The set of assumptions modified to apply to COTS service history are as follows:

1. Defects are repaired immediately when they are discovered.
2. Defect repair is perfect.
3. No new code is introduced during the time period of service history.
4. All reported defects are from similar usage.
5. Each unit of execution time is equally likely to find defects.
6. Service history accurately represents the proposed operational profile.

United States Navy Submarine Electronic System Acquisition Program Offices (Team Submarine), Strategy 2000, Commercial Off-The-Shelf (COTS) Acquisition Primer, 12 November 1996

This is an acquisition guide for the U.S. Navy. The document states that the use of item history and previous user experience can and should be used for determining maintenance aspects of the system. The document does not make a distinction between software and hardware for this point; the discussion is at the system level. However, the notion of accurate prediction of errors, maintenance needs, and environmental factors can be extended to software.

Judith A. Clapp and Audrey E. Taub, "A Management Guide to Software Maintenance in COTS-Based Systems," MP 98B0000069, MITRE Corporation, November 1998

"For some complex products with a large number of commercial users, there may be a user's group. The program can have representation in the group to help influence the vendor to correct deficiencies and improve the product. One of the often-cited advantages of COTS products is the large number of users who help to find errors so that the quality of the product improves. This does lead to a phenomenon that the maintenance organization must contend with; namely, that the vendor will send out a stream of patches to the software to fix errors found by other users. The maintainer must decide which to incorporate, and when. The fixes may be in areas of the COTS package that are not used by the system, or could potentially lead to other errors because of the way that the system uses the COTS package. The maintainers must find a low-risk way of dealing with these patches. One of the difficult problems in maintaining the quality of the system when a COTS software product is replaced is how much to test. Clearly, only black box testing can be done. When the product is replaced, it is important to test for any side effects that may not be directly at the interface of the product with the rest of the system. Collecting problem reports and analyzing them is an important part of quality control for COTS software. The data should show when the problem was due to an error in a COTS product and how long it took for a correction. This kind of information can show trends in the reliability of each product and the kind of service the maintainer of that product is giving. These metrics are useful in determining when a product needs to be replaced because of its poor performance or that of its vendor."

Points to note:

1. Do user groups influence the vendor to repair what is needed by one group of people rather than allowing safety-critical errors to be fixed first?

2. Some users may choose not to implement the patch—does the problem-reporting mechanism track different users with different versions?

3. When users implement a patch that was meant to fix someone else's problems, it may result in more problems because of differences in the usage of the same software. How does the vendor track this phenomenon as a error resulting from error fix?

4. The author recommends reliability measurement of the product using the error metrics.

Ljerka Beus-Dukic, "Non-Functional Requirements for COTS Software Components," School of Computing and Mathematics, University of Northumbria at Newcastle

This paper has a good discussion of what the author calls nonfunctional requirements, which may be especially important for products that are built for general use. One of the major points made in this paper is the domain differences and that "Vendors" vested interest in a particular domain (if sufficient it provides a scope for introducing component features relevant to that domain)." The nonfunctional requirements are the "ilities" that are important to that user or that domain: maintainability, availability, integrity, usability, portability etc. These requirements often escape comparisons of what is required in the proposed environment. This paper agrees with NATO arguments that when a reuse is brought from one domain to other care should be exercised to make sure that the domain differences are accounted for.

Shane Lunga and Meera Galoria, "Using COTS Components: Their Location, Qualification and Selection," DERA Portsdown West, UK

This paper does not have much information on service history. However, one point of interest is the in-service life support and evolvability which highlights some of the qualities that one may put in the contract with the vendor including the compatibility with its previous version and to a limited extent with its next version. The paper has a good set of criteria for selection of COTS.

W.M. Gentleman, "If Software Quality is a Perception, How Do We Measure It?" Software Engineering Laboratory, National Research Council, Canada

This is a very good, thought-provoking article on quality. "In the past, much of the thinking about quality has been in the context of one-on-one customer/contractor relationship. Galsworthy (1912) extolled the virtues of hand crafted custom products. Today's customer often faces a very different situation: a choice between competitive off-the-shelf products." This article celebrates the value of "field experience" above system requirements. The need for new types of metrics to understand and analyze off-the-shelf products is proposed. The author recognizes that we are moving to qualitative measures with this shift in usage of field experience.

"Software quality is often defined in terms of the fitness of the product for its purpose." This definition of quality stretches what we are trying to do with service history. Service history is used to compute error rates so that we have a good idea of what to expect in the future use of the system in a "similar" purpose. How different will it be from the purpose for which it was used during the service history? "... objective measures should be used to support subjective assessment."

C. Jones, R.E. Bloomfield, P.K.D Froome, and P.G Bishop, "Methods for Assessing the Safety Integrity of Safety-Related Software of Uncertain Pedigree (SOUP)," Adelard, Contract Research Report 337/2001, 2001

This is an excellent document on software of uncertain pedigree. It also has a good comparison of various standards. There are many good points made in general for COTS. The most interesting quote from this document is, "The PDS should be provided with documentation

equivalent to the rest of the safety-related software. However, in cases where the PDS is justified on the basis of in-service history or extensive V&V and is treated as a "black box," it may be acceptable for design information not to be provided as long as a comprehensive requirement specification for the software is provided." There is an analysis to show what kind of errors can be found by black box testing and what other complementary methods may be used to complete V&V of the product whose in-service history is known. The document agrees with DO-178B in requiring that the following data be used in the computation of error rates:

- The length of the service period
- The operational hours, allowing for different operational modes, and the numbers of copies in service
- The definition of what is counted as a fault/error/failure

P.G. Bishop, R.E Bloomfield, and P.K.D Froome, "Justifying the Use of Software of Uncertain Pedigree (SOUP) in Safety-Related Applications" Adelard, Contract Research Report 336/2001, 2001

An excellent document, which is a companion to the other Adelard document on SOUP. Asserts that "(field experience) Gives feedback on failures occurring in field operation. Applicable to SOUP, but needs to be of high quality to demonstrate reliability." The document has been written to analyze how a SOUP can satisfy the requirements of IEC 61508; a similar analysis would be useful for guiding COTS compliance to DO-178B.

Scott, J.A., Preckshot, G.G., and Gallagher, J.M., "Using Commercial Off-The-Shelf (COTS) Software in High-Consequence Safety Systems," Lawrence Livermore National Laboratory, UCRL-JC-1222246, November 1995

This document has a lot of insight into how to ensure COTS in safety-critical systems. One similarity between this document and the Adelard documents is the reliance on standards for different characteristics of the COTS. This document ties all of the requisite qualities of COTS with IEEE standards. There are many ideas that stem out of the discussions in the document.

The document discusses dissimilarities of installations between sites; although the same product is being used. This point may be of interest when many copies of the subject software are running in different installations during the service history duration.

The other idea is to require that the problems be tracked.

There is a discussion of values of gradations of assurance for different levels of safety.

a. For the highest level of safety, Category A, the suggestion is the following: "The COTS product should have significant (> 1 year operating time), current severe-error-free operating experience in at least two independent operating locations. Adverse reports should not be excluded even if two operating locations can be found with no adverse reports. The version and release of the proposed COTS product should be identical to that

used in the experience database. The configuration of the product in the experience data base should closely match that of the proposed COTS product."

b. For the next, Category B: "The COTS product has operated satisfactorily in similar applications. The version and release of reported experience may not be identical to the proposed COTS product, but a consistent configuration management program and well-managed update program provide traceability and change control. Error reporting, tracking, and resolution are consistent and correctly attributable to version and release. The version and release proposed has no major unresolved problems. A current bug list should be available to COTS purchasers as a support option."

c. For Category C: "The COTS product has been shown to operate without serious malfunction in the instant application. An error-reporting scheme is planned or in place that tracks malfunctions of this COTS product in applications controlled by this applicant. Documentation and records retention allow error histories of 5 years or length of service, whichever is shorter."

Jim Krodel, "Commercial Off-The-Shelf (COTS) Avionics Software Study," United Technologies Research Center on contract for the Federal Aviation Administration, DOT/FAA/AR-01/26, May 2001

This document contains a discussion of what an applicant or a regulator should be mindful of when service history is proposed for certification credit. "It seems intuitive that operating experience data derived from extensive usage of the same version of a product in similar applications would indicate that the product is acceptable for the intended application. Several issues with this assumption should be considered. First, configuration data of the actual version used in the problem report supplied can be difficult to obtain. As such, the statistical validity of the data is unknown. Error-reporting databases commonly span multiple releases and configurations of a given product. Another issue is that circumstances surrounding the occurrence, monitoring, and recording of failures are often vaguely reported. Even the highest avionics software level system have difficulty in reproducing and diagnosing problems due to the inability to recreate the problem from the information provided. Indeed, some of these activities are not only improperly controlled, but in fact, there might be some motivation for COTS vendors to limit publication of negative experiences, particularly if the COTS product was not originally intended for safety-critical systems. Another key issue regarding operating experience is that extensively used products can still have crucial faults that could cause problems in safety systems. The tendency is to consider operational experience to be like extensive random testing. In this regard, operational experience suffers from the same shortcoming as testing: testing cannot prove the absence of faults."

A.2 DATA COLLECTION AND SYNTHESIS 2: INTERVIEWS.

A.2.1 INTERVIEWEES.

Interviewees included engineers who have worked on verification and validation of systems in the nuclear domain, utility companies, DoD in general, the U.S. Navy, and general safety-critical application to a variety of application domains. Interviews for the most part focused on

regulations and practice; some conversations were also had on conclusions reached because of personal experiences.

A.2.2 INTERVIEW CHECKLIST.

A basic set of questions for interviews was asked to assess the use of service history in other domains of industry (nuclear, consumer industry, chemical industry, DoD etc.).

The actual questions were slightly modified during the conversations, depending upon the domain and the interviewee's field of specialization.

1. Introduction: A brief synopsis of what we are doing. The term service history may not be used in the domain at hand. Straighten out the vocabulary of COTS, service history, certification, assurance, equivalent level of safety, and alternate means.

2. Scope: Software whose development process is not known, may be COTS or other previously developed software.

3. Service history may be used as a means of establishing confidence in the integrity of software (COTS or for grandfathering of software). Is this means allowed in your industry domain? By itself or in conjunction with other means or both? Is use through explicit or implicit means?

    a. What regulations are in place? Are copies of these regulations available? Are they well understood by the regulators, by the industry? Are there instances where the interpretation of regulations differed? Explain in what areas, and how they were resolved?

    b. How long have these regulations been used?

    c. In your opinion, what are its good points, its bad points?

    d. How would you improve these regulations?

4. If this is an ad hoc process, what is the usual process? Is it widely understood?

    a. What are its advantages and disadvantages?
    b. How would you improve the process whether formal or ad hoc?

5. Whatever method is used, how does the applicant provide assurance for the following questions?

    a. Time: How long is long enough? How did you derive the number? Is it based on engineering judgment or software reliability measures?

    b. Operation: How do you know that the operation in question is exactly the same as during the service history?

c. Environment: Data collected in one environment can be used to gain confidence in another environment only if the environments are similar. How do you assess/establish similarity of environment? How do you assure that mode confusion problems are resolved? How do you assure that differences in kind of exceptions, and exception handling are resolved? How do you assure that human factors issues such as expected behavior of the system under similar circumstances are resolved?

d. Problem Reporting: Many of the COTS suppliers do not have a rigorous problem reporting system. If credit were taken for service history duration, one would have to assure that all of the problems were reported, and all of them were logged and dealt with. How do you assess the quality of problem reporting and configuration of software?

6. Is any of the above assurance activities graduated by the criticality of the system in which service history is to be applied? In avionics applications, the assurance needed depends upon the software criticality, which is a measure of the damage that may be caused if the software were to fail.

7. In assuring avionics, the usual process involves assuring the development process and assuring the product by verification.

   a. How do you address proof of adequacy of development process commensurate with the safety and reliability expected?

   b. What methods do you use to assure the product? Is safety assured by your methods for COTS using service history equivalent to if you had overseen the development process? In this question, service history may have been used by itself or in conjunction with other measures.

8. Are there particular examples that you can share regarding uses of service history by itself or in conjunction with other measures in your domain. Please also state the level of safety and reliability expected of these examples.

9. In your opinion have these been success stories?

10. What lessons have been learned from these cases? What additional measures would you recommend in the future assurance projects involving COTS? In particular regarding service history?

11. Is there anyone else in your organization who you would suggest talking to?

12. Any one else in related fields or suppliers who might contribute to this study?

A.2.3 INTERVIEW RESULTS.

Interviews 1 and 2:

In the Nuclear domain, Title 10 Part 50 regulations are followed for safety-critical systems that are called Class 1E. These systems are built very conservatively—mostly hardware (HW) and firmware and analog systems. They have to be physically robust (such as earthquake proof) and redundant. There is not much chance of using COTS in this class of systems.

All of the other software are divided into different classes in different power plants. Nuclear Software Management Group is the industry group for software management issues to be resolved. There is a graded approach to building software. At the highest levels of safety the vendor may be required to be "appendix B qualified" which means that the QA system is of the standard that is expected by the NRC through Title 10 part 50 requirements. At lower levels of safety, contracts are used to assure that proper error reporting is established for both safety and business purposes. Corporate administration process is also used for reporting problems. These records are usually kept for the life of the code. There may be some records that are kept for the life of the power plant plus 10 years.

Service experience of COTS or reuse software is of interest only as a risk mitigation tool. Not much credence is given to this data. Testing for intended functions is the way to gain assurance of software. When the testing is successful, commercial dedication is achieved-i.e., although the part or software is not built for the nuclear application, the testing has shown that the part or software is high grade.

Advantages of this type of "informal" approach are that progress may be made quickly. The disadvantages are the risks associated with any informal process. In the nuclear industry there is a pact to alert fellow power plant businesses of safety problems, and share this data freely. At the national level the Institute for Nuclear Power Operations (INPO) and at the world level World Association of Nuclear Operators (WANO) are used as vehicles for maintaining the high level of safety in the nuclear field.

Federal regulations are used as the minimum that an operator would have to do; many of the power plants do more than what is required by the regulations. It is a matter of safety and prudent business decisions.

Interview 3:

Domain: DoD and Nuclear

Applications:
1. SW development tools
2. COTS used within Mission Planning software and Mapping software

Roger noted mixed experience in case of development tools. Sometimes the tools did not perform as advertised. Also, service history was very thin since these were the cutting edge tools. The problems noted were due to scale of application; the other successful applications of COTS may have been of small scale. Complexity is also an issue. He would advocate using

complexity and size as criteria for whether to use COTS. Three to five years of service experience for a tool such as a compiler would be acceptable. Beyond 5 years the problem is that the tool is loosing its edge in technology. Quality of service history or problem data records available from the vendor is varied depending upon the vendor.

In mission planning and mapping software, simpler COTS have been used with success. Again, these are simpler uses than the ones described for tools. Complexity is an issue. Roger would not advocate using COTS within software systems of high criticality. For the kind of applications that he is talking about, capabilities, not service history forms the basis for whether to buy those COTS. Size (scalability) is not an issue in these particular types of software (mission planning or mapping). Ease of use, ease of connectivity, and ease of openness are important. A copy of COTS software is tested in a prototype before making the buy decision. Mode is an issue that is hard to address. Usually requirements are obtained from the vendor and the system is tested thoroughly using these requirements. If one vendor has the requirements and another does not for an otherwise comparable product, Roger would buy the product with requirement with the hope that the vendor has better software engineering practices. Testing will be done to exercise all features. Roger has again noticed differences in the details of problem reporting between vendors. In-house problem reporting kept at his company trace the problems down to the LRU level so that when time comes to upgrade the COTS, a decision can be made with respect to the vendor. In-house reports log the resolutions also. Roger emphasized that for critical applications he would not recommend COTS. Roger would like to see COTS vendors using software engineering principles so that their product would be of better quality.

Interview 4:

Domain: U.S. Navy, General safety-critical applications

There is no absolute need to use COTS in all cases; when the safety of the system is in question, it is not clear that equivalence to development assurance can be gained by using service history.

Interview 5:

In nuclear applications a process of assurance of safety critical software whose development history is not known, is documented in IEEE ANS 7-4.3.2. This document assigns what is known as “commercial dedication” to a piece of equipment that may then be used in safety critical applications. The assurance depends upon intense product verification not just on product service history. Product service history is of interest only in a qualitative sense.

A.3 DATA COLLECTION AND SYNTHESIS 3: INDUSTRY BREAKOUT SESSION.

During the FAA National Software Standardization Conference in Danvers, MA (June 5-7, 2001) a breakout session was conducted to solicit industry opinion and ideas on use of service history for certification credit. Twenty-five engineers from industry, as well as government attended the session. The following data was collected based on the comments made during the discussion. The comments validated some of the conclusions independently reached by literature search. The comments also provided some new ideas such as use of tool qualification to justify software reliability method.

A.3.1 COMMENTS CAPTURED DURING THE SESSION.

Questions of Time

Primary flight control (PFC) was initially approved to DO-178B, Level B. The company wanted to bring it over from one aircraft to another. The question was how to combine multiple sets of service history data along with existing design assurance data from the previous software approval.

Can time measurement cut across environments and operations at all?

What is the appropriate unit of time? The warm and fuzzy factor - more qualitative than quantitative.

Statistical significance must be extrapolated down from the aircraft level to a system contribution to a particular failure condition.

How do you separate out the true software contributions from the hardware failure rates, infant mortality, etc?

Taxonomy may be hiding considerations of the "big" picture

Are there certain objectives that simply may not be met through service history?

It may be that service history must be decoupled from individual objectives – you're really making an argument against the FARs/JARs

HOWEVER, it is likely that the most common cases where people would like to employ service history arguments is just that, namely credit for one or more specific objectives (e.g. service history of x hours for MC/DC)

If you have a failure that is attributable to software but can clearly show through analysis that it is not safety-related, you don't have to restart the clock. [CNS/ATM group]

What about the problem of problems introduced by modding the code?

Determination of the amount of time will be, by definition, a case-by-case basis - too many variables to define a one size fits all solution.

Success story: Ten years of data, non-flight-critical, three well-defined updates, level B - with an independent monitor (would have been level A otherwise), approx 280 A/C with complete service hours documented (approx 75K)

Bottom-line, however, was the argument was made to FARs not specific objectives, i.e., no tit for tat arguments

How do you take into account functions that operate for only a small window of the total flight hours (e.g., 150 sec for WAAS)

Whatever you do here, it must be relevant and significant!

What about framing this in terms of events?

DOE - Agricultural domain to extrapolate time - similar to HALT

Questions of Operation

How much exposure has each mode seen? Is it ever possible to have visibility to this information - what would the metric look like?

What happens when you translate a common mode of operation from the civil arena to the military arena (e.g. FMC pt A to pt A)

Approach this from the perspective of a trade study. Are there different weightings for the various questions? Weighting criteria tend to be subjective. Can it be tied back to safety? Isn't this just reliability in disguise?

Consider an exponential curve with a lot of latitude at level D - probably not at all at level A

State analysis, human factors, or other approaches to get at differences between the two operational domains - this is the antithesis of the dissimilarity argument

Questions of Environment

Consider any changes in hardware, interfacing systems, etc

Everything in terms of changes must tie back to safety assessment.

Don't forget to consider the ramifications of processor changes, compiler changes, linkers, etc...

What about just a "minor" change to the option set used to build (e.g. optimization)

Questions of Problem Reporting (CM)

Is the failure going to be reportable (visible) to allow a problem report even to be generated?

Problem versus a feature?

Attribute the problem to what?

What do you do with the "one-offs" (nonrepeatable, one-time occurrences)?

All PRs must go through safety analysis process.

Military environment - encourage problem reporting under warranty, discourage after warranty expires due to costs born by unit

NFF problems

Trace-back and fault isolation are key to ensuring the integrity of data.

In-service PR's are much more important than development PRs for the purposes of evaluating maturity (i.e. legitimate service history) Suggested to consider about 2 -3 months prior to release.

MTTR for software problems - lots of considerations in how you get at this actual metric.

Is there an industry average that can be derived for particular types of aircraft to predict expected numbers of problems to externally validate what is being seen?

What is the effect of "operational" workarounds on this whole scenario?

Drop off of reporting once problem becomes known.

Conclusions as presented at the conference:

- Multiple time duration based on available life cycle data and criticality
- Statistical significance derived from the aircraft level to the software level
- FAR/JAR safety argument instead of DO-178B objectives
- Measurement of time must consider exposure to the function and be stated in appropriate units
- Must be relevant and significant (as specified in DO-178B)
- Clock must be restarted for failures having a safety implication
- Artificially extending the duration by statistical prediction
- Approach service history from the perspective of a trade study – give different weights to different attributes
- Service history is really the antithesis of the dissimilarity argument in DO-178B
- Many problems are simply not visible in a way that allows appropriate problem reporting (attributable to software)
- All problem reports should be investigated through safety assessment
- Problem reporting may be encouraged or discouraged for business, procedural or perception reasons
- Nonrepeatable and "no fault found" problems – what then?
- Can Mean Time To Repair (MTTR) be used as a measure of the severity of the problem?

APPENDIX B—SERVICE HISTORY—LITERATURE SEARCH

About 200 references were examined for this project. Not all of the references had a discussion on service history. There was a contradiction inherent to commercial software that embraced COTS because of the novelty of technology. No consideration was given to service history data and, in many cases, none existed. The only domains where service history played an important role were safety-critical domains of nuclear, defense, and aviation. Of these fields, it was generally found that the European literature is more prolific in addressing the use of service history. Further, the European safety culture also accepts software reliability measures. The close connection between software reliability and product service history helps the discussion of both the topics in European defense standards, guidelines for medical devices, and nuclear applications.

B.1 GENERAL PAPERS.

This category includes general material that spans multiple industry segments or is generic to the topic of COTS or service history. In addition, this section contains material that relates to the use of COTS and service history in consumer products such as those evaluated by Underwriter's Laboratories (UL).

Many of the COTS used in aviation (mainly ground systems and less frequently in avionics) have a product history in the consumer domain (cars, telecommunications, display, and sound from entertainment systems, etc.). Further, considerations in using product history for product assurance are common across the various domains.

B.1.1 Bibliography.

1. "Commercial Off-The-Shelf (COTS) Software Acquisition Guidelines and COTS Policy Issues," *Communications and Information Systems Agency,* NATO, 10 January 1996.

2. Abts, Christopher, "COTS Software Integration Cost Modeling Study," Technical Report, University of Southern California, Center for Software Engineering, Barry Boehm, Director. The report (along with other information about COCOTS) is available at: http://sunset.usc.edu/COCOTS/cocots.html.

3. Anderson, Evan, "A Heuristic for Software Evaluation and Selection," *Software Practice and Experience*, Vol. 19, No. 8, August 1989, pp. 707-717.

4. Balk, D.L. and A. Kedia, "PPT:A COTS Integration Case Study," T. Rowe Price Investment Technologies, *Proceedings from the 22nd International Conference on Software Engineering (ICSE)-22*, 2000, Limerick, Ireland.

5. Beach, J.R., and M.L. Bonewell, "Setting-Up a Successful Software Vendor Evaluation/Qualification Process for 'Off-the-Shelve' Commercial Software Used in Medical Devices," *Proceedings of the of Sixth Annual IEEE Symposium on Computer-Based Medical Systems*, Ann Arbor, MI, June 1993, IEEE Computer Society, Los Alamitos, California, pp. 284-288.

6. Belton, Valerie, "A Comparison of the Analytic Hierarchy Process and a Simple Multi-Attribute Value Function," *European Journal of Operational Research*, Vol. 26, No. 1, 1986, p. 7-21.

7. Besnard, J.F., S.J. Keene, and J.M. Voas, "Assuring COTS Products for Reliability and Safety Critical Systems," *Proceedings of the Annual Reliability and Maintainability Symposium,* Washington, DC, January 1999, IEEE, Piscataway, NJ, pp. 317-322.

8. Beus-Dukic, L., and A. Wellings, "Requirements for a COTS Software Component: A Case Study," Conference on European Industrial Requirements Engineering (CEIRE '98), London, England, October 1998, *Requirements Engineering*, Vol. 3, No. 2, Springer-Verlag, pp. 115-120.

9. Boehm, B., B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "Cost Models for Future Software Life Cycle Processes: COCOMO 2.0," *Annals of Software Engineering,* Vol. 1, Baltzer (Amsterdam, 1995) pp. 57-94.

10. Boehm, B., B. Clark, E. Horowitz, C. Westland, R. Madachy, and R. Selby, "The COCOMO 2.0 Software Cost Estimation Model: A Status Report," *American Programmer*, Vol. 9, No. 7, July 1996, pp. 2-17.

11. Boehm, B. and H. In, "Identifying Quality-Requirement Conflicts," *IEEE Software*, Vol. 13, No. 2, March 1996, pp. 25-35.

12. Boehm, B.W., "Improving Software Productivity," *23rd IEEE Computer Society International Conference COMPCON Fall 81. Productivity an Urgent Priority*, Washington, September 1981, IEEE, New York, pp. 2-16.

13. Boloix, Germinal and Pierre Robillard, "A Software System Evaluation Framework," *IEEE Computer,* December 1995, pp. 17-26.

14. Braun C.L., "A Lifecycle Process for the Effective Reuse of Commercial Off-The-Shelf (COTS) Software," GTE Information Systems, SSR '99, Los Angeles, California, USA.

15. Briand, L.C., "COTS Evaluation and Selection," *Proceedings of the International Conference on Software Maintenance,* Bethesda, Maryland, November 1998, IEEE Computer Society, Los Alamitos, pp. 222-223.

16. Brown, A. and K. Wallnau, "A Framework for Evaluating Software Technology," *IEEE Software,* Vol. 13, No. 5, September 1996, pp. 39-49. Also "A Framework for Systematic Evaluation of Software Technologies," Component-Based Software Engineering, IEEE Computer Society, Los Alamitos, California, 1996, pp. 27-40. (http://www.sei.cmu.edu/cbs/papers/paper1.ps).

17. Brownsword, L., "Lessons Learned Applying Commercial Off-The-Shelf Products," COTS-Based Systems Initiative, Carnegie Mellon University, Software Engineering Institute, Technical Note CMU/SEI-99-TN-015, 1999.

18. Brownsword, L., D. Carney, and Oberndorf, "The Opportunities and Complexities of Applying Commercial Off-The-Shelf Components," Carnegie Mellon University, Software Engineering Institute, April 1998.

19. Cárdenas-García, Sergio and Marvin Zelkowitz, "A Management Tool for Evaluation of Software Designs," *IEEE Transactions on Software Engineering*, Vol. 17, No. 9, September 1991, pp. 961-971.

20. Carney, D., "COTS Evaluation in the Real World," Carnegie Mellon University, Software Engineering Institute, December 1998.

21. Carney, D., "Requirements and COTS-Based Systems: A Thorny Question Indeed," Carnegie Mellon University, Software Engineering Institute, June 1999.

22. Carney, D., "COTS Product Evaluation and System Design," Carnegie Mellon University, Software Engineering Institute, March 1999.

23. Carney, D., "Evaluation of COTS Products: Some thoughts on the Process," Carnegie Mellon University, Software Engineering Institute, September 1998.

24. Carney, David J. and Kurt C. Wallnau, "A Basis for Evaluation of Commercial Software," *Information and Software Technology,* 1998, pp. 851-860.

25. Copigneaux, Frédéric, "Guide for Measurement, Rating and Assessment 2: Buyer's Guide," Draft Version 2, *Technical Report of ISO/IEC JTC1/SC7/WG6*, 1991.

26. COTS information and white papers from National Research Council, Canada.

27. Dick, R. and R. Hunter, "Subjective Software Evaluation," *Software Quality Management II-Building Quality Into Software 2,* 1994, 776, pp. 321-334.

28. Discussion of COTS and COTS selection considerations on Carnegie-Mellon Software Engineering Institute web page http://www.sei.cmu.edu/cbs/papers/eval_bib.html.

29. Discussions of Software Service History in the Safety Critical news group, July 2000. (safety-critical@cs.york.ac.uk).

30. Embedded Software Task Team, "Lessons Learned on Embedded Software Developmental Programs—A Guidebook for Program Managers," Boeing, March 1995.

31. Expert Choice, Inc., Battelle Memorial Institute, "Protest Proof Source Selection," October, 1998. (http://www.expertchoice.com/selection/).

32. Finkelstein, A., G. Spanoudakis, and M. Ryan, 'Software Package Requirements and Procurement," *Proceedings of the 8th International Workshop on Software Specification and Design,* Schloss Velen, Germany, March 1996, IEEE Computer Society Press, Los Alamitos, California, pp. 141-145.

33. Fox, Greg, Karen Lantner, and Steven Marcom, "A Software Development Process for COTS-Based Information System Infrastructure," *Proceedings of the Fifth International Symposium on Assessment of Software Tools,* 1997, pp. 133-142.

34. Frakes, W., R. Prieto-Diaz, and C. Fox, "DARE-COTS. A Domain Analysis Support Tool," *Proceedings of the 17th International Conference of the Chilean Computer Science Society,* Valparaiso, November 1997, IEEE Computer Society Los Alamitos, California, pp. 73-77.

35. Helander, M.E., Ming Zhao, and N. Ohlsson, "Planning Models for Software Reliability and Cost," *IEEE Transactions on Software Engineering*, Vol. 24, No. 6, June 1998, pp. 420-434.

36. Heller, Rachelle S., "Evaluating Software: A Review of the Options," *Computers and Education,* Vol. 17, 1991, pp. 285-291.

37. Herschel, D., "Techniques for Selecting a Start-Up Vendor," *Gartner Group*, #TG-06-0532, 1998, 3 pages.

38. Hissam, S., "Correcting System Failure in a COTS Information System," *Proceedings of the International Conference on Software Maintenance*, November 1998.

39. Hissam, S. and D. Carney, "Isolating Faults in Complex COTS-Based Systems," *SEI Monographs on the use of Commercial Software in Government Systems*, Carnegie-Mellon Software Engineering Institute, February 1998.

40. IEE Colloquium on COTS and Safety Critical Systems (Digest number 1997/013), IEE, 1996.

41. IEEE/IEA 12207-1 Guide for Information Technology.

42. Interview with FDA on use of Service History in certification of medical devices.

43. Interview with UL on use of Service History in certification of consumer electronics.

44. ISO/IEC, Information Technology—Software Product Evaluation—Part 5: Process for Evaluators, ISO/IEC 14598-5 (1998). (http://webstore.ansi.org/AnsiCatalog/product.asp?sku=26983)

45. ISO/IEC, Information Technology—Software Product Evaluation, ISO/IEC 9126 (1991-12-15). (http://webstore.ansi.org/AnsiCatalog/product.asp?sku=27267).

46. ITN/FBI (Identification and Tasking Networking/Federal Bureau of Investigation), *COTS Evaluation and Qualification Plan,* 1995, pp. 1-22.

47. Jeanrenaud, A. and P. Romanazzi, "Software Product Evaluation Metrics: Methodological Approach," *Software Quality Management II—Building Quality into Software 2* (1995) 776, pp. 59-69.

48. Judith A. Clapp and Audrey E. Taub, "A Management Guide to Software Maintenance in COTS-Based Systems," MP 98B0000069, MITRE Corporation, November 1998.

49. Jung, Ho-Won and Byoungju Choi, "Optimization Models for Quality and Cost of Modular Software Systems," *European Journal of Operational Research*, Vol. 112, No. 3, February 1999, pp. 613-619.

50. Karlsson, J. "Software Requirements Prioritizing," *Proceedings of the Second International Conference on Requirements Engineering,* Colorado Springs, CO, April 1996, IEEE Computer Society Press, Los Alamitos, California, pp. 110-116.

51. Kitchenham, B.A., S.G. Linkman, and D.T. Law, "Critical Review of Quantitative Assessment," *Software Engineering Journal*, Vol. 9, No. 2, March 1994, pp. 43-53.

52. Kitchenham, B., S. Linkman, and D. Law, "DESMET: A Methodology for Evaluating Software Engineering Methods and Tools," *Computing & Control Engineering Journal,* Vol. 8, No. 3, June 1997, pp. 120-126.

53. Kitchenham, Barbara, Lesley Pickard, and Shari Lawrence Pfleeger, "Case Studies for Method and Tool Evaluation," *IEEE Software Features,* Vol. 12, 1995, pp. 52-62.

54. Klopping, Inge M. and Christopher F. Bolgiano. "Effective Evaluation of Off-The-Shelf Microcomputer Software," *Office Systems Research Journal,* Vol. 9, 1990, pp. 46-50.

55. Kontio, Jyrki, "A Case Study in Applying a Systematic Method for COTS Selection," *Proceedings of the 18th International Conference on Software Engineering* 1996, pp. 201-209.

56. Kontio, Jyrki, "OTSO: A Systematic Process for Reusable Software Component Selection," Technical Report CS-TR-3378, UMIACS-TR-95-63, University of Maryland College Park, Maryland, 20742, December 1995.

57. Kropp, N.P., P.J. Koopman, and D.P. Siewiorek, "Automated Robustness Testing of Off-The-Shelf Software Components," Digest of Papers, *28th Annual International Symposium on Fault-Tolerant Computing*, Munich, June 1998, IEEE Computer Society, Los Alamitos, California, pp. 230-239.

58. Lanubile, F., and G. Visaggio, "Evaluating Predictive Quality Models Derived From Software Measures: Lessons Learned," *Journal of Systems and Software*, Vol. 38, No. 3, September 1997, pp. 225-234.

59. Lichota, Randall W., Robert L. Vesprini, and Bruce Swanson, "PRISM Product Examination Process for Component Based Development," *Proceedings of the Fifth International Symposium on Assessment of Software Tools,* 1997, pp. 61-69.

60. Ljerka Beus-Dukic, "Non-Functional Requirements for COTS Software Components," School of Computing and Mathematics, University of Northumbria at Newcastle.

61. Lubars, M., C. Potts, and C. Richter, "A Review of the State of the Practice in Requirements Modeling," *Proceedings of IEEE International Symposium on Requirements Engineering,* San Diego, CA, January 1993, IEEE Computer Society Press, Los Alamitos, California, pp. 2-14.

62. Maiden, N.A., and C. Ncube, "Acquiring COTS Software Selection Requirements," *IEEE Software*, Vol. 15, No. 2, March/April 1998, pp. 46-56.

63. Maiden, N.A., C. Ncube, and A. Moore, "Lessons Learned During Requirements Acquisition for COTS Systems," *Communications of the ACM*, Vol. 40, No. 12, December 1997, pp. 21-25.

64. Maiden, N.A.M., and G. Rugg, "ACRE: Selecting Methods for Requirements Acquisition," *Software Engineering Journal*, Vol. 11, No. 3, May 1996, pp. 183-192.

65. Mayrand, Jean and François Coallier, "System Acquisition Based on Software Product Assessment," *Proceedings of the 18th International Conference on Software Engineering* 1996, pp. 210-219.

66. McDougall, Anne and David Squires, "A Critical Examination of the Checklist Approach in Software Selection," *Journal of Educational Computing Research,* December 1995, pp. 263-274.

67. Miller, James R. and Robin Jeffries, "Usability Evaluation: Science of Trade-Offs," *IEEE Software Real-Time Realities,* 1992, pp. 97-102.

68. Min, Hokey, "Selection of Software: The Analytic Hierarchy Process," *International Journal of Physical Distribution and Logistics Management*, Vol. 22, No. 1, 1992, pp. 42-52.

69. Miyazaki, Y. and K. Mori, "COCOMO Evaluation and Tailoring," *Proceedings of the 8th International Conference on Software Engineering,* London, England, August 1985, IEEE Computer Society Press, Washington, D.C., pp. 292-299.

70. Morisio, M. and A. Tsoukiàs, "IusWare: A Methodology for the Evaluation and Selection of Software Products," *IEEE Proceedings of Software Engineering*, Vol. 144, No. 3, June 1997, pp. 162-174.

71. Morisio, M., C. Seaman, A. Parra, V. Basili, S. Kraft, and S. Condon, "COTS-Based Software Development: Processes and Open Issues, Journal of Systems and Software, to appear December 2001.

72. Morisio, M., C. Seaman, A. Parra, V. Basili, S. Condon, and S. Kraft, "Investigating and Improving a COTS-Based Software Development Process," University of Maryland, *Proceedings of the 22nd International Conference on Software Engineering (ICSE)-22*, 2000, Limerick, Ireland.

73. Ncube, C., "PORE: The Procurement Oriented Requirements Engineering Method," Centre for HCI Design, School of Informatics, City University, London, 1998. (http://www.soi.city.ac.uk/pore/).

74. Office of Management of the Budget, "Evaluating Information Technology Investments," March 1999. (http://www.whitehouse.gov/WH/EOP/OMB/infotech/infotech.html)

75. Palvia, Prashant, "A Comprehensive Model and Evaluation of the Software Engineering Environment," *IRMA Conference Proceedings,* 1992, pp. 302-307.

76. Powell, Antony, "A Practical Strategy for the Evaluation of Software Tools," *International Federation for Information Processing* (IFIP), 1996, pp. 165-185.

77. Puma Systems, Inc., "Commercial Off-The-Shelf System Evaluation Technique (COSSET)," web: http://www.pumasys.com/cosset.htm, prior to 25 March 1999, 5 pages.

78. Robert, Philippe, "Guide to Software Product Evaluation—The Evaluator's Guide," (ISO/IEC 9126-5), Technical Report ISO/IEC JTC1/SC7 N1136, July 20, 1993.

79. Schneidewind, N.F., "Methods for Assessing COTS Reliability, Maintainability, and Availability," *Proceedings of the International Conference on Software Maintenance,* Bethesda, Maryland, November 1998, IEEE Computer Society, Los Alamitos, pp. 224-225.

80. Sha, L., Goodenough, and Pollak, "Simplex Architecture: Meeting the Challenges of Using COTS in High-Reliability Systems," Software Engineering Institute, CROSSTALK, April 1998.

81. Shane Lunga and Meera Galoria "Using COTS Components: Their Location, Qualification and Selection," DERA Portsdown West, UK.

82. Shaw, Mary, "Truth vs. Knowledge: The Difference Between What a Component Does and What We *Know* It Does," *Proceedings of the 8th International Workshop on Software Specification and Design*, March 1996.

83. Shin, Hyunsik and Jinjoo Lee. "A Process Model of Application Software Package Acquisition and Implementation," *Journal of Systems and Software* Vol. 32 (1996), pp. 57-64.

84. Stark, G. E., "Estimating the Number and Severity of Software Defects for Staffing, Release and Risk Decision-Making," Tivoli, Software Technology Conference, May 2001.

85. The Safety-Critical Systems Club Newsletter, January 2001, Volume 10, Number 2, University of Newcastle upon Tyne, UK.

86. The Safety-Critical Systems Club Newsletter, May 2001, Volume 10, Number 3, University of Newcastle upon Tyne, UK.

87. Tran, V., Dar-Biau Liu, and B. Hummel, "Component-Based Systems Development: Challenges and Lessons Learned," *Proceedings of the Eighth IEEE International Workshop on Software Technology and Engineering Practice Incorporating Computer Aided Software Engineering*, London, July 1997, IEEE Computer Society, Los Alamitos, California, pp. 452-462.

88. Tran, Vu and Dar-Biau Liu, "A Risk-Mitigating Model for the Development of Reliable and Maintainable Large-Scale Commercial Off-The-Shelf Integrated Software Systems," *Proceedings Annual Reliability and Maintainability Symposium* (1997) pp. 361-367.

89. Tran, Vu, and Dar-Biau Liu, "A Procurement-Centric Model for Engineering Component-Based Software Systems," Proceedings of the Fifth International Symposium on Assessment of Software Tools, 1997, pp. 70-79.

90. Vigder, M.R. and J.C. Dean, "Building Maintainable COTS-Based Systems," *Proceedings of the International Conference on Software Maintenance*, Bethesda, MD, November 1998, IEEE Computer Society, Los Alamitos, California, pp. 132-138.

91. Voas, J., "Error Propagation Analysis for COTS Systems," *Computing & Control Engineering Journa*l, Vol. 8, No. 6, December 1997, pp. 269-272.

92. Welzel, Dieter and Hans-Ludwig Hausen, "A Method for Software Evaluation With Respect to Quality Standards," *Proceedings of the 3rd International Conference on Achieving Quality in Software*, Chapman & Hall, 1996, pp. 381-399.

93. Williams, Fred, "Appraisal and Evaluation of Software Products," *Journal of Information Science*, Principles and Practice 18, 1992, pp. 121-125.

B.2 NUCLEAR.

Although their use of software is fairly small compared to civil aviation, the nuclear industry, and in particular the Nuclear Regulatory Commission (NRC), invested significant funds in establishing guidelines for the use of COTS in nuclear applications. These guidelines along with the associated research efforts have established ground rules including many elements of service history.

Both the aviation sector and the nuclear sector are concerned with safety. The NRC follows a tiered system of three levels of safety criticality. The aviation industry may be able to draw upon the experiences in other regulated safety-critical industry.

B.2.1 BIBLIOGRAPHY.

1. G.G. Preckshot and J.A. Scott "A Proposed Acceptance Process for Commercial Off-The-Shelf (COTS) Software in Reactor Applications," Lawrence Livermore National Laboratory, Prepared for U.S. Nuclear Regulatory Commission, NUREG/CR-6421 UCRL-ID-122526.

2. Gallagher, John, (ed), "Discussions Related to COTS Obtained From Expert Peer Review Meeting on High Integrity Software for MIPPs Conducted by MITRE for NRC Research," May 24-26, 1994.

3. Interview with Bob Brill and John Gallagher of NRC regarding use of Service History in the Nuclear Sector.

4. Lawrence Livermore National Laboratory, UCRL-ID-122526, March 1996.

5. Lawrence, D.J., "Workshop on Developing Safe Software: Final Report," Lawrence Livermore National Laboratory, UCRL-ID-113438, November 1992.

6. Lawrence, D.J., W.L. Parsons, and G.G. Preckshot, "Evaluation Software for Safety Systems in Nuclear Power Plants," Lawrence Livermore National Laboratory, UCRL-JC-116038, April 1994.

7. Preckshot, G.G. and J.A. Scott, "Vendor Assessments and Software Plans," Lawrence Livermore National Laboratory, UCRL-ID-122243, November 1995.

8. Preckshot, G.G. and J.A. Scott, "A Proposed Acceptance Process for Commercial Off-The-Shelf (COTS) Software in Reactor Applications."

9. Proceedings of the Digital Systems Reliability and Nuclear Safety Workshop, NUREG/CP-0136, March 1994.

10. Scott, J.A. and J.D. Lawrence, "Testing Existing Software for Safety Related Applications," Lawrence Livermore National Laboratory, UCRL-ID-117224, September 1995.

11. Scott, J. A., G.G. Preckshot, and J. M. Gallagher, "Using Commercial Off-The-Shelf (COTS) software in High-Consequence Safety Systems," Lawrence Livermore National Laboratory, UCRL-JC-1222246, November 1995

12. Sheper, C.O., "Certification of Reusable Software Components," Technical Report, Rome Laboratory, March 1993

13. Sparkman, D., "Techniques, Processes, and Measures for Software Safety and Reliability," Lawrence Livermore National Laboratory, UCRL-ID 108725, May 30, 1992.

B.3 DEFENSE.

Much of the data that has been collected for review as the research plan is developed originates in the defense sector. It is important to note at this early stage that the relevance and usefulness of this data has not been established. This caveat stems from the difference in focus of mission critical versus safety critical.

Many of the COTS used in DoD are used in military aviation. Rules for use of service history within this COTS selection and acquisition process may help answer some of the concerns within the civil aviation community.

B.3.1 Bibliography.

1. "Buying Commercial Off-The-Shelf," Draft Air Force Handbook, Acquisition Management-Standardization, 1999.

2. "SD-2 Buying Commercial and Nondevelopment Items: A Handbook," Office of the Under Secretary of Defense for Acquisition and Technology, December 1998.

3. "Software Product Re-Use and COTS Software," Joint MoD/Industry Computing Policy.

4. Abts, C., "COTS Software Integration Cost Modeling Study," University of Southern California, for USAF ESC/Hanscom, June 1997.

5. Bergmann, Walter B., II, (approved by signature), "Buying Commercial and Non-developmental Items: A Handbook," Office of the Under Secretary of Defense for Acquisition and Technology, 1996, 97 pages.

6. Carney, David "Quotations from Chairman David: A Little Red Book of Truths to Enlighten and Guide in the Long March toward the COTS," Written for SEI Joint Program Office, Hanscom AFB, Carnegie-Mellon Software Engineering Institute, July 1998.

7. COTS Journal, The RTC Group.

8. CROSS TALK magazines, The Journal of Defense Software Engineering.

9. Demko, E., "Commercial Off-The-Shelf (COTS)—A Challenge to Military Equipment Reliability," Annual Reliability and Maintainability Symposium, Las Vegas, NV, January 1996.

10. DY 4, "From Avionics to Vetronics: Considerations for Application of COTS VME to Deployed Defense Systems," http://www.dy4.com/cots/position.htm.

11. DY 4, "Harsh Environments COTS Handbook," http://www.dy4.com/cots/hdbk.htm.

12. Federal Acquisition Regulations, Parts 10-12, 1996.

13. Hissam, S., Carney, D., and Plakosh, D., "DoD Security Needs and COTS-Based Systems," *SEI Monographs on the Use of Commercial Software in Government Systems*, Carnegie-Mellon Software Engineering Institute, September 1998.

14. Interview with AF personnel regarding use of Service History in acquisition of Avionics with COTS.

15. Joint Services Computer Resources Management Group, "Software System Safety Handbook," December 1999.

16. Mil-HDBK-1221, "Evaluation of Commercial Off-The-Shelf (COTS) Manuals," August 1995.

17. Oberndorf, Patricia and Carney, David, "A Summary of DoD COTS-Related Policies," *SEI Monographs on the Use of Commercial Software in Government Systems,* Carnegie-Mellon Software Engineering Institute, September 1998.

18. Title 10, United States Code, Ch 140 (Sections 2375-2377), "Procurement of Commercial Items," 1996.

B.4 CIVIL AVIATION AND SPACE APPLICATIONS.

Service History has been an area of debate ever since the rules for civil aircraft certification were established. References in this area cover the latest dialogues underway in RTCA SC-190 on the subject, as well as prior work and guidance. A small number of sources have been identified from the space sector to supplement this area.

Use of COTS in Aviation raises concerns of safety, reliability and product obsolescence from both the industry and regulators. Further, there are concerns of measurable objective evidence that can be applied to variety of products built using variety of processes. Concerns need to be addressed in any policy that will be useful for both the regulators and the industry.

B.4.1 Bibliography.

1. "NASA Preferred Reliability Practices, Part 1 of 3: Design and Test Practices for Aerospace Systems," NASA/TM-4322, National Aeronautics and Space Administration, February 2000.

2. "Software Safety," NASA-STD-8719-13A, National Aeronautics and Space Administration, September 1997.

3. "Use of COTS/NDI in Safety-Critical Systems," Report of the Challenge 2000 Subcommittee of the FAA Research, Engineering, and Development Advisory Committee, February 1996.

4. Civil Aviation Authority Publication 670 SW 01, issue 3. Requirements for Software Safety Assurance in Safety Related ATS Equipment. Available by request from A. Eaton, Safety Regulation Group, UK Civil Aviation Authority, Aviation House, Gatwick Airport South, West Sussex, England, RH6 0YR.

5. C. Jones, R.E Bloomfield, P.K.D Froome, and P.G Bishop, "Methods for Assessing the Safety Integrity of Safety-Related Software of Uncertain Pedigree (SOUP)," Adelard, Contract Research Report 337/2001, 2001.

6. Coombes, A.C., "Report to Civil Aviation Authority Safety Regulation Group, Comparison of Standards for Safety Related Software Development," YSE Reference: CF 17 1/3/53, 25 February 1999.

7. DeWalt, M., "Level D and COTS white paper," Draft, 1997.

8. Federal Aviation Administration, "Guidelines for the Approval of Software Changes in Legacy Systems Using RTCA DO-178B," Notice N 8110.89.

9. Federal Aviation Administration, "Guidelines for the Oversight of Software Change Impact Analyses Used to Classify Software Changes as Major or Minor," Notice N 8110.85

10. Gerlich, R., "Lessons Learned by Use of (C)OTS," Proceedings of the DASIA 98—Data Systems in Aerospace (SP-422), Athens, May, 1998, ESA, Paris, France, pp. 517-523.

11. Hayhurst, K. J., Dorsey, C. A., Knight, J. C., Leveson, N. G., McCormick, F. G., "Streamlining Software Aspects of Certification: Report on the SSAC Survey," NASA/TM-1999-209519, National Aeronautics and Space Administration, August 1999.

12. Jim Krodel, "Commercial Off-The-Shelf (COTS) Avionics Software Study," United Technologies Research Center on contract for the Federal Aviation Administration, DOT/FAA/AR-01/26, May 2001.

13. Nuselbeh B., "ARIANE 5: Who Dunnit," *IEEE Software*, May/June 1997, pp 25-16.

14. P.G. Bishop, R.E. Bloomfield, and P.K.D Froome, "Justifying the Use of Software of Uncertain Pedigree (SOUP) in Safety-Related Applications" Adelard, Contract Research Report 336/2001, 2001.

15. Presentation materials from the Safety-Critical Systems Club seminar "COTS and SOUP: Current Thinking and Work in Progress," Newcastle University, UK, April 2001.

16. Prof. J.L. Lions, "ARIANE 5 Flight 501 Failure," Report by the Inquiry Board, Paris, 19 July 1996.

17. RTCA SC 167 Papers on Service History, Previously Developed Software and Alternate Means.

18. RTCA SC 167, "Software Considerations in Airborne Systems and Equipment Certification," document number RTCA/DO-178B, RTCA, Inc., December 1992.

19. RTCA SC 180, "Design Assurance Guidance for Airborne Electronic Hardware," document number RTCA/DO-254, RTCA Inc., April 2000.

20. RTCA SC 189, "Guidelines for Approval of the Provision and Use of Air Traffic Services Supported by Data Communications," document number RTCA/DO-264, RTCA Inc., December 2000.

21. RTCA SC 190, papers on Service History, Previously Developed Software and Alternate Means.

22. RTCA SC 190, Web discussions on Service History, Previously Developed Software and Alternate Means.

23. RTCA SC 190, "Second Annual Report for Clarification of DO-178B," document number RTCA/DO 248A, RTCA Inc., September 2000.

24. Sharp, D.C., "Reducing Avionics Software Cost Through Component Based Product Line Development," Software Technology Conference Proceedings, April 1998.

25. Struck, W., "Guidance for Assessing the Software Aspects of Product Service History of Airborne Systems and Equipment," Discussion Paper P-17, Certification Authorities Software Team, Federal Aviation Administration.

26. Winter, D.C., "Modular, Reusable Flight Software for Production Aircraft," 15th AIAA/IEEE Digital Avionics Systems Conference Proceedings, October 1996, pp. 401-406.

B.5. SUPPLEMENTAL—SOFTWARE TESTING, RELIABILITY, AND SAFETY.

The four areas of software testing, risk management, reliability, and safety were included in the literature search to aid in the assessment of service history approaches as they relate to certification. Given the breadth of each of these areas, this part of the search was more narrowly defined to those sources that would allow determination of how service history relates to, could be established by, or could be assessed through their use.

B.5.1 BIBLIOGRAPHY.

1. "Reviewer Guidance for Computer Controlled Medical Devices Undergoing 510(k) Review," Office of Device Evaluation, Center for Devices and Radiological Health, Food and Drug Administration, August 1991.

2. AIAA/ANSI, "Software Reliability Recommended Practice," AIAA/ANSI R-013-1992, American Institute of Aeronautics and Astronautics (AIAA), 1992.

3. Brocklehurst, S. and Littlewood, B., "New Ways to Get Accurate Reliability Measures," *IEEE Software,* July 1992, pp. 34-42.

4. Commission Electrotechnique Internationale International Electrochemical Commission (IEC) 61508 "Functional Safety: of Electrical/Electronic/Programmable Electronic Safety-Related Systems," 1997.

5. Councill, W.T., "Third Party Testing and the Quality of Software Components," *IEEE Software,* July/August 1999, pp. 55-57.

6. Dylis, D.D. and Priore, M.G., "A Comprehensive Reliability-Assessment Tool for Electronic Systems," Reliability and Maintainability Symposium Proceedings, IEEE Reliability Society, 2001.

7. Ehrlich, W., Prasanna, B., Stampfel, J., and Wu, J., "Determining the Cost of a Stop-Test Decision," *IEEE Software,* March 1993, pp. 33-42.

8. Everett, W.W., "Reliability and Safety of Real-Time System," *IEEE Software,* May 1995, pp. 13- 27.

9. Garlan, D., Allen, R., and Ockerbloom, J., "Architectural Mismatch or Why It's Hard to Build Systems Out of Existing Parts," *17th International Conference on Software Engineering*, 1995.

10. Goel, A.L., "Statistical Methods and Software Engineering: A Mini Tutorial," *27th Symposium on the Interface: Computing Science and Statistics*, Pittsburgh, PA, June 21-24, 1995.

11. Gokhale, S.S., P.N. Marinos, and K.S. Trivedi, "Effect of Repair Policies on Software Reliability," Lucent Technologies, Bell Laboratories, IEEE, COMPASS, 1997.

12. Graves, T.L., Karr, A.P., Marron, J.S., and Sly, Harvey, "Predicting Fault Incidence Using Software Change History," *IEEE Transactions on Software Engineering*, Vol. 26, No. 7, July 2000.

13. Hecht, H., and M. Heckt, "Qualitative Interpretation of Software Test Data," SoHar, Inc.

14. Hecht, M. and H. Heckt, "Use of Importance Sampling and Related Techniques to Measure Very High Reliability Software," IEEE Aerospace 2000 Conference, Big Sky, MT, March 2000.

15. Hecht, M., D. Tang, and H. Hecht, "Software Reliability Assessment-Myth and Reality," SoHar, Inc., 1996.

16. Hecht, M., H. Hecht, and D. Tang, "Quantitative Reliability and Availability Assessment for Critical Systems Including Software," SoHar Inc., IEEE, COMPASS, 1997.

17. Herrmann, D.S., "*Software Safety and Reliability,"* IEEE Computer Society, 1999.

18. Jones, Capers, "Applied Software Measurement," second edition, Computing, McGraw-Hill, 1996.

19. Kanoun, K., M. Kaaniche, and J. Laprie, "Qualitative and Quantitative Reliability Assessment," *IEEE Software*, March/April 1997, pp. 77-87.

20. Kontio, J., "A Case Study in Applying a Systematic Method for COTS Selection," University of Maryland, Proceedings from the 19th International Conference on Software Engineering, ICSE-19, 1997.

21. Krolio, A., A. Fritz, and B. Bertsche, "Correlation Between Failure Behaviour of Automotive Components Under Taxi and Field Operating Conditions," Reliability and Maintainability Symposium Proceedings, IEEE Reliability Society, 2001.

22. Kohl, R.L., "Establishing Guidelines for Suitability of COTS for a Mission Critical Application," AverStar, Inc.

23. Lawrence, D.J., "Software Reliability and Safety in Nuclear Reactor Protection Systems," Lawrence Livermore National Laboratory, UCRL-ID-114839, June 1993.

24. SAE G-11 SW, "Software Reliability Program Standard," SAE JA 1002, July 1998.

25. Kuball S., J. May, and G. Hughes, "Building a System Failure Rate Estimator for Identifying Component Failure Rates," in 10th International Symposium on Software Reliability Engineering (ISSRE '98), pages 32-41 IEEE Computer Society, November 1999.

26. Wood Alan, "Software Reliability Growth Models: Assumptions vs. Reality." Proceedings of the 8th International Symposium on Software Reliability Engineering (ISSRE "97).

27. Lawrence, D., "An Overview of Software Safety Standards," Lawrence Livermore National Laboratory, UCRL-JC- 122249, November 1995.

28. Leveson, N.G., *Safeware*, Addison-Wesley Publishing Company, 1995.

29. Liu, D., "A Risk Mitigation Model for Large-Scale COTS Integrated Software Systems," Proceedings of the Annual Reliability and Maintainability Symposium, 1999.

30. Lyu, M., and A. Nikora, "Applying Reliability Models More Effectively," *IEEE Software,* July 1992, pp. 43-52.

31. Michael R. Lyu, ed., *Handbook of Software Reliability Engineering,* IEEE Computer Society Press, 1996.

32. Mil Std 882D, "Standard Practice for System Safety," Department of Defense, February 2000.

33. Ministry of Defence, Defence Standard 00-40 (Part 6)/Issue 1 (ARMP-6) "Reliability and Maintainability, Part 6: In-Service R & M," December 1988.

34. Ministry of Defence, Defence Standard 00-43 (Part 1)/Issue 1, "Reliability And Maintainability Assurance Activity, Part 1: In-Service Reliability Demonstrations," January 1993.

35. Ministry of Defence, Defence Standard 00-44 (Part 1)/Issue 2, "Reliability And Maintainability, Data Collection and Classification, Part 1: Maintenance Data and Defect, Reporting in the Royal Navy, the Army, and the Royal Air Force," June 1995.

36. Ministry of Defence, Defence Standard 0055, "Requirements for Safety-Related Software in Defense Equipment Part 1 and Part 2," August 1997.

37. Ministry of Defence, Defence Standard 0056, "Safety Management Requirements for Defence Systems, Part 1 and Part 2," December 1996.

38. Musa, J., "Operational Profiles in Software Reliability Engineering," *IEEE Software,* March 1993, pp 14-32.

39. Musa, J., *Software Reliability Engineering,* McGraw-Hill, 1998.

40. Poore, J. H., H. D. Mills, and D. Mutchler, "Planning and Certifying Software System Reliability," *IEEE Software*, January 1993, pp. 88-99.

41. Rodriguez-Dapena, P., "Software Safety Certification: A Multidomain Problem," *IEEE Software,* July/August 1999, pp. 31-38.

42. Tausworthe, R.C. and M. Lyu, "A Generalized Technique for Simulating Software Reliability," *IEEE Software*, March 1996, pp. 77-88.

43. United States Navy submarine electronic system acquisition program offices (Team Submarine), Strategy 2000, "Commercial Off-The-Shelf (COTS) Acquisition Primer," 12 November 1996.

44. Voas, J., "Certification: Reducing the Hidden Costs of Poor Quality," *IEEE Software,* July/August 1999, pp. 22-25.

45. Voas, J., "Certifying Software for High-Assurance Environments," *IEEE Software,* July/August 1999, pp. 48-54.

46. W.M. Gentleman, "If Software Quality Is a Perception, How Do We Measure It?" Software Engineering Laboratory, National Research Council, Canada.

47. W.M. Gentleman "Effective Use of COTS (Commercial Off-The-Shelf) Software in Long Lived Systems," Software Engineering Laboratory, National Research Council, *Proceedings From the 19th International Conference on Software Engineering, ICSE-19*, 1997, Boston, Massachusetts, USA.

48. W. M. Gentleman, "Architecture for Software Construction by Unrelated Developers," *Software Architecture, TC2 First Working IFIP Conference on Software Architecture (WICSAI)*, February 1999.

49. Wakid, S.A., Kuhn, R., and Wallace, D.R., "Toward Credible IT Testing and Certification," *IEEE Software,* July/August 1999, pp. 39-47.

50. Wallace, D.R., L.M. Ippolito, and R.D. Kuhn, "High Integrity Software Standards and Guidelines," NIST Special Publication 500-204, National Institute of Standards and Technology, September 1992.

51. Weyuker, E.J., "Testing Component-Based Software: A Cautionary Tale," *IEEE Software,* September/October 1998, pp. 54-59.

52. Xie, M., G.Y. Hong, and C. Wohlin, "A Practical Method for the Estimation of Software Reliability Growth in the Early Stage of Testing," *Proceedings of the 8th International Symposium on Software Reliability Engineering (ISSRE '97)*, 1997.