

DRAFT

Handbook for Object-Oriented Technology in Aviation (OOTiA)

Volume 4: Certification Practices

vPC.0

January 30, 2004



This Handbook does not constitute Federal Aviation Administration (FAA) policy or guidance, nor is it intended to be an endorsement of object-oriented technology (OOT). This Handbook is not to be used as a standalone product but, rather, as input when considering issues in a project-specific context.

Contents

- 4.1 PURPOSE..... 1
- 4.2 BACKGROUND..... 1
- 4.3 ACTIVITIES FOR STAGES OF INVOLVEMENT..... 1
 - 4.3.1 *Activities for Stage of Involvement #1 – PLANNING REVIEW*..... 1
 - 4.3.2 *Activities for Stage of Involvement #2 – DEVELOPMENT REVIEW*..... 3
 - 4.3.3 *Activities for Stage of Involvement #3 – VERIFICATION/TEST REVIEW*..... 7
 - 4.3.4 *Activities for Stage of Involvement #4 – FINAL REVIEW*..... 9
- 4.4 REFERENCES 10

Tables

<i>Table 4.3-1 SOI 1 Activities/Questions</i>	3
<i>Table 4.3-2 SOI 2 Activities/Questions</i>	7
<i>Table 4.3-3 SOI 3 Activities/Questions</i>	9
<i>Table 4.3-4 SOI 4 Activities/Questions</i>	10

4.1 Purpose

Volumes 1-3 of the OOTiA Handbook have pointed out a number of potential issues and some ways to address the issues, when using OOT in aviation products. Because OOT poses a number of potential certification issues, it may require additional oversight by certification authorities and their designees, as well as the applicants. This volume provides an approach for certification authorities and designees to ensure that OOT issues have been addressed in the projects they are reviewing and/or approving.

4.2 Background

In 1998 the FAA's Software Review Job Aid [2] for conducting software reviews prior to certification was released. The Job Aid was intended to provide a standardized approach for certification authorities and designees to perform software reviews in order to assess compliance to DO-178B objectives. In 2003 the Job Aid was updated to address lessons learned and to correct some typographical errors. Additionally, Chapter 2 of the FAA Order 8110.49, "Software Approval Guidelines," addresses the software review process [1]. The focus of the Job Aid is a series of activities and questions to be addressed by reviewers to ensure that the project meets DO-178B objectives.

Since the software review (either on-site or desk reviews) is the primary means for certification authorities and designees to assess compliance to DO-178B objectives, this volume provides a series of activities and questions to be considered during a software review for object-oriented projects. The activities and questions are in a format similar to the Job Aid and may eventually be implemented into the Job Aid. The activities and questions are divided among the four stages of involvement (SOI) that are discussed in the Job Aid and Order 8110.49. Each activity has a set of questions designed to help carry out that activity. It is important to realize that these questions are not all encompassing. Some questions will not be applicable to all OOT projects and some projects will require additional questions. Each question includes reference(s) to DO-178B and the other OOTiA Handbook volumes.

4.3 Activities for Stages of Involvement

4.3.1 *Activities for Stage of Involvement #1 – PLANNING REVIEW*

Section 2-4.a of Order 8110.49 states: *The software planning process is the initial process in the software life cycle for any software project. The planning process establishes the various software plans, standards, procedures, activities, methods, and tools required to develop, verify, control, assure, and produce the software life cycle data. The intent of the software planning review is to determine if the applicant's plans and standards provide an acceptable means for satisfying the objectives of RTCA/DO-178B. This review can also reduce the risk of an applicant producing a software product that does not meet RTCA/DO-178B objectives or other certification criteria [1].*

The primary goals of SOI 1 are to:

- Ensure that plans meet objectives in DO-178B Annex A tables A-1, A-8, A-9, A-10;
- Assess that when the applicant follows their plans, they will meet all of the objectives of DO-178B (Tables A-1 to A-10).

When OOT is used in a project, the Activities/Questions shown in Table 1 below should be considered, **in addition to those in the Job Aid for SOI 1**. If the plans and standards indicate that a number of OOT issues exist in the project and are not being addressed, an issue paper may be needed for the specific project.

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
1.1	Review all software plans and consider the following questions:		

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
1.1.1	Does the additional considerations section (or some other section) of the Plan for Software Aspects of Certification address that the project is object-oriented (OO) and identify OO-related issues for the specific project?	2.3.4	A1: 4
1.1.2	Have the issues raised in volume 2 of the handbook been addressed by the applicant's plans? Note: Volume 3 provides some approaches for addressing OO issues; however, applicants may have other means to address the issues.	Section 2.3	All
1.1.3	<p>Do the plans address how the following common OO development issues will be addressed?</p> <ul style="list-style-type: none"> • Encapsulation • Overloading • Inheritance • Dynamic binding/dispatch • Polymorphism • Templates • Inlining • Dead/deactivated code • Traceability • Structural coverage 	<p>Section 2.3</p> <p>(See also Volume 3)</p>	A1: 4
1.1.4	Do the plans define how the OO life cycle data maps to the DO-178B Section 11 life cycle data? For example, what are the high-level requirements, low-requirements, design, and source code in OO?	2.3.1.1	A1: 1-4 (Sect 11.1e)
1.1.5	What levels of abstraction are used for the OO implementation and how does the abstraction map to the DO-178B objectives (reference 2.3.1.1 of Volume 2)?	2.3.1.1	A1: 1
1.1.6	How are derived requirements addressed and tracked?	2.3.1.2	A1: 1
1.1.7	Has the difference between dead and deactivated code for OO been clarified in the planning phases of the program?	2.3.2.4.1	A1: 4
1.1.8	If a modeling language is used (e.g., UML), how is the software functionality described and how is the tie to the safety assessment established?	2.3.1.2	A1: 1
1.2	Review the Development Standards (requirements, design, & coding standards) and consider the following questions:		
1.2.1	Do the development standards address limitations for OO implementation (e.g., addressing rules that are not supported by the target language)?	2.3.1.1 and 2.3.1.2	A1: 5 (Sect 11.6-11.7)

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
1.2.2	Are OO methods and notations documented in the software requirements and design standards? Specifically, do the standards identify how non-functional requirements are captured, and how low-level and derived requirements are addressed?	2.3.1.1 and 2.3.1.2	A1: 5 (Sect 11.6- 11.7)
1.2.3	Do the development standards implement the solutions to the OO issues as the PSAC (or other plans) claim they will?	N/A	A1: 5 & 7
1.2.4	Do the standards disallow “constructs or methods that produce outputs that cannot be verified or that are not compatible with safety-related requirements” (per DO-178B, section 4.5c)?	2.3.1.3	A1: 5 (Sect 4.5c)
1.2.5	Are necessary restrictions for the chosen OO language identified in the coding standards?	2.3.1.3	A1: 5 (Sect 11.8)
1.2.6	If constructs or methods were disallowed, do the plans address how proper implementation will be verified?	2.3.1.3	A1: 1,2,5,&7
1.3	Determine if an Issue Paper (IP) is needed for the project. An IP is typically required if the plans do not address the known issues (e.g., the issues presented in Volume 2).	N/A	Varies

Table 4.3-1 SOI 1 Activities/Questions

4.3.2 Activities for Stage of Involvement #2 – DEVELOPMENT REVIEW

Section 2-5.a of Order 8110.49 states: *The software development processes are the software requirements, design, code, and integration processes. The development processes are supported by the integral processes of software verification, configuration management, quality assurance, and certification liaison processes. Therefore, the software development review should assess the effective implementation of the applicant’s plans and standards through examination of the software life cycle data, particularly the software development data and integral processes’ data associated with it. During this review, the applicant and FAA may agree on and document changes to or deviations from plans and standards discovered during the review. Before conducting a software development review, the software development data should be sufficiently complete and mature*[1].

The purpose of SOI 2 is to:

- Assess effective implementation of applicant’s plans and standards through examination of software life cycle data;
- Assess and agree to any changes in the plans;
- Assure that software life cycle data meets DO-178B objectives from tables A-2, A-3, A-4, A-5, A-8, A-9, and A-10.

When OOT is used in a project, the Activities/Questions shown in Table 2 below should be considered, **in addition to those in the Job Aid for SOI 2.**

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
2.1	Assess the developer’s traceability approach by considering the following questions:		
2.1.1	How is traceability between requirements, design, code, and test cases/procedures established and maintained?	2.3.3.4	A3:6 A4:6 A5:5
2.1.2	Does the traceability approach address common OO traceability problems, such as: <ul style="list-style-type: none"> • traceability of functional requirements through implementation might be lost or difficult with an OO program because of mismatches between function-oriented requirements and an object-oriented implementation • complexity of class hierarchies may cause traceability problems • tracing through OO views - a key concern is that behavior of the classes and how they interact together to provide the required function might not be visible in any single view. 	2.3.3.4	A3:6 A4:6 A5:5
2.1.3	Does the traceability approach provide a way to ensure that: <ul style="list-style-type: none"> • all requirements (both high and low-level) are verified, • unintended functions and/or source code are identified, and • visibility into derived requirements exists? 	2.3.3.4	A3:6 A4:6 A5:5
2.1.4	Do the OO tools/methods support traceability across the full life cycle and across multiple views?	2.3.3.4	A3:6 A4:6 A5:5
2.2	Determine if sub-type issues have been addressed by considering the following questions:		
2.2.1	How has the applicant addressed type substitutability? For example, has Liskov Substitution Principle (LSP) been implemented?	2.3.2.1	A2, A3, & A4
2.2.2	Have inconsistent type uses been addressed?	2.3.2.1.2	A2, A3, & A4
2.2.3	If a language sub-set is proposed, does it address the sub-type issues?	2.3.2.1	A2, A3, & A4
2.3	Determine if subclass issues have been addressed by considering the following questions:		

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
2.3.1	<p>If multiple inheritance is used, is the intent of subclass operations clear and unambiguous? Consider the follow questions to consider if the issues have been addressed:</p> <ul style="list-style-type: none"> • If the same operation is inherited by an interface via more than one path through the interface hierarchy (repeated inheritance), is it clear whether this results in a single operation in the subinterface or in multiple operations? • Is the complete definition/intention of a class clear (in order to avoid a subclass being incorrectly located in a hierarchy)? • How is the class hierarchy accuracy assured? • If a subinterface inherits different definitions of the same operation [as a result of redefinition along separate paths], is it clear whether/how they should be combined in the resulting subinterface? • Do top-heaviness or deep hierarchies exist (six or more subclasses)? If so, how is it verified that the wrong variable type, variable, or method are not inherited? • How are unintended connections among classes (which could lead to difficulty in meeting the DO-178B/ED-12B objective of data and control coupling) prohibited? • How is the subinterface merging addressed when <i>different</i> parent interfaces define operations with different names but compatible specifications? • How are “name clashes” avoided, when more than one parent <i>independently</i> defines an operation with the same signature? • If multiple interface inheritance is used, is the developer’s intent unambiguous? 	2.3.2.2.1	A3: 1-4, 6,7 A4: 1-4, 6-11
2.3.2	<p>Have overriding concerns been addressed? Specifically:</p> <ul style="list-style-type: none"> • Can an operation accidentally override another? • Can operations inherited from different sources be accidentally joined? • Can a subclass-specific implementation of a superclass be accidentally omitted? • Have the five classes of errors associated with overriding that were identified in Volume 2 (section 2.3.2.2.2) been addressed? 	2.3.2.2.2	A3: 1-4, 6,7 A4: 1-4, 6-11, 13
2.4	If objects or design components are being reused, have dead/deactivated code issues been addressed? Specifically:		
2.4.1	Are the issues of dead and deactivated code addressed consistently throughout the project and as the planning documents stated?	2.3.2.4.1	A5 & A7 (Sect 5.4.3 & 6.4.4.3)
2.4.2	Does overriding result in any dead or deactivated code?	2.3.2.4.1	A5 & A7 (Sect 5.4.3 & 6.4.4.3)

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
2.4.3	Do libraries and OO frameworks result in dead or deactivated code?	2.3.2.4.2	A5 & A7 (Sect 5.4.3 & 6.4.4.3)
2.5	Does the OO approach address the following configuration management issues:		
2.5.1	Is configuration management maintained when objects and classes are used multiple times in slightly different manners?	2.3.3.3	A8
2.5.2	If a modeling tool is used, how is the configuration of objects and classes handled?	2.3.3.3	A8
2.5.3	How is configuration control maintained for inherited classes, association links, and client relationships?	2.3.3.3	A8
2.6	If tools are used, consider the following questions:		
2.6.1	Does the tool support compliance with the DO-178B objectives? I.e., is the tool compatible with DO-178B?	2.3.4.1	Objs vary depending on the tool (Sect 12.2)
2.6.2	How mature is the tool (e.g., compiler, UML tool, ...) that is being used? Unstable configuration may indicate an immature tool.	2.3.4.2	Objs vary depending on the tool (Sect 12.2)
2.6.3	How will the tool be maintained to meet the long-term needs of an aircraft project?	2.3.4.2	A8: 6 (Sect 7.2.9)
2.6.4	If a tool does not meet the DO-178B definition of development or verification tool, how is it addressed? I.e., Are new classes of tools being used? If so, is an issue paper needed? Does the tool need to be qualified?	2.3.4.3	Objs vary depending on the tool (Sect 12.2)
2.6.5	If the tool is being qualified, does it meet DO-178B and Order 8110.49 guidelines?	2.3.4.3	Objs vary depending on the tool (Sect 12.2)
2.7	Consider if the following additional issues have been addressed:		

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
2.7.1	Is garbage collection used? If so, how is determinism established?	2.4	A2, A3, A4, A5
2.7.2	Is exception handling used? If so, how are control flow, run-time support, determinism, and deactivated code addressed?	2.4	A2, A3, A4, A5
2.7.3	How is concurrency address? Use of this feature raises issues with control flow, run-time support, real-time predictability, and deactivated code.	2.4	A2, A3, A4, A5
2.7.4	How is any additional complexity managed in OO projects?	2.4	All
2.7.5	If libraries are used, do they meet DO-178B objectives?	2.3.2.4.2	All

Table 4.3-2 SOI 2 Activities/Questions

4.3.3 Activities for Stage of Involvement #3 – VERIFICATION/TEST REVIEW

Section 2-6.a of Order 8110.49 states: *The software verification process is typically a combination of inspections, demonstrations, reviews, analyses, tests, and coverage analysis. As with the other reviews, the software configuration management and quality assurance processes are also active during these verification activities. The verification activities confirm that the software product specified is the software product built. Therefore, the software verification review should ensure that the software verification processes will provide this confirmation and will result in objective evidence that the product has been sufficiently tested and is the intended product. The purpose of the software verification review is to: assess the effectiveness and implementation of the applicant's verification plans and procedures; ensure the completion of all associated software configuration management and quality assurance tasks; ensure that the software requirements, design, code, and integration have been verified; and ensure that the software verification process will achieve the requirements-based test coverage and structural coverage criteria of RTCA/DO-178B, Annex A, Table A-7. Before conducting a software verification review, the software verification process should be sufficiently complete and mature[1].*

The purpose of SOI 3 is to:

- Assess effective implementation of the applicant's verification plans and procedures;
- Check the completion of all associated software configuration management and quality assurance tasks;
- Make determination on acceptable deviations from plans and standards found during the review or with the applicant's requested deviations;
- Ensure that the selected software requirements have been verified;
- Ensure that the software life cycle data meets DO-178B objectives from Tables A-2, A-6, A-7, A-8, A-9, and A-10;
- Ensure that the verification activity satisfied the structural coverage requirements found in DO-178B, Table A-7.

When OO is used in a project, the Activities/Questions shown in Table 3 below should be considered, **in addition to those in the Job Aid for SOI 3.**

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
--------	------------------------------	-----------------	-------------

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
3.1	Consider if data and control coupling issues have been addressed by considering the following questions:		
3.1.1	Does the OO approach address data and control coupling analysis?	2.3.3.1.1	A7: 8
3.1.2	If dynamic dispatch is used, how is data and control coupling analysis performed?	2.3.3.1.1	A7: 8
3.1.3	If inlining is used, how is data and control coupling analysis performed?	2.3.3.1.1	A7: 8
3.2	Determine if memory management and initialization issues have been addressed by considering the following questions:		
3.2.1	Does the memory allocation/deallocation approach result in predictable worst-case memory analysis?	2.3.2.3	A6 (Sect 6.4.3a)
3.2.2	Is memory fragmentation and defragmentation handled in a deterministic manner?	2.3.2.3	A6 (Sect 6.4.3a)
3.2.3	Have memory leaks been addressed?	2.3.2.3	A6 (Sect 6.4.3a)
3.2.4	Have initialization problems been identified and addressed?	2.3.2.3	A6 (Sect 6.4.3a)
3.3	Consider if structural coverage issues have been addressed by considering the following questions:		
3.3.1	If dynamic dispatch is used, how is structural coverage addressed?	2.3.3.1.2	A7: 5-7
3.3.2	If inheritance is used, how is structural coverage addressed?	2.3.3.1.2	A7: 5
3.3.3	If polymorphism is used, how is structural coverage addressed?	2.3.3.1.2	A7: 5
3.3.4	If inlining is used, how is structural coverage addressed?	2.3.3.1.2	A7: 5
3.3.5	If templates are used, how is structural coverage addressed?	2.3.3.1.2	A7: 5
3.3.6	If source to object code traceability is required (per DO-178B, section 6.4.4.2b), how is it carried out? Specifically, how do dynamic dispatch, inlining, and type conversion affect source to object code traceability?	2.3.3.1.4	A7: 7
3.4	Have the effects of dynamic dispatch and inlining on stack usage and timing analysis been addressed?	2.3.3.1.3	A5: 6

Item #	Evaluation Activity/Question	OOTiA Vol 2 Ref	DO-178B Obj
3.5	Consider if the testing issues have been addressed by considering the following questions:		
3.5.1	Has requirements-based testing, software integration testing, and hardware/software integration testing been properly performed? The key concern for requirements testing is that the mapping of function-oriented test cases to an object-oriented implementation might not be obvious since the basic unit of testing in an OO program is not a function or a subroutine, but an object or a class.	2.3.3.2.1	A6
3.5.2	Has test coverage of both high and low level requirements been completed? Also, test coverage of high-level and low-level requirements will likely require different testing strategies and tactics from the traditional structured approach because information hiding and abstraction techniques decrease or complicate the observability of low-level functions.	2.3.3.2.1	A7: 3, 4
3.5.3	Has a retest or reuse of test cases approach been implemented to address inheritance and overriding? The concern to be addressed is: Requirements-based testing is complicated by inheritance, dynamic dispatch, and overriding because it might be difficult to determine how much testing at a superclass level can be reused for its subclasses.	2.3.3.2.2	A6 & A7

Table 4.3-3 SOI 3 Activities/Questions

4.3.4 Activities for Stage of Involvement #4 – FINAL REVIEW

Section 2-7.a of Order 8110.49 states: *The final software build establishes the configuration of the software product considered by the applicant to comply with all objectives of RTCA/DO-178B. It is the version of the software intended to be used in the certified system or equipment. The purpose of this review is to: determine compliance of the final software product with the appropriate objectives of RTCA/DO-178B; ensure that all software development, verification, quality assurance, configuration management, and certification liaison activities are complete; ensure a software conformity review has been completed; and review the final Software Configuration Index (SCI) and Software Accomplishment Summary (SAS). The final certification software review should take place when the software project is completed*[1].

The purpose of SOI 4 is to:

- Determine if final compliance to all of the DO-178B objectives has been achieved and any open items addressed/dispositioned;
- Assess the Software Configuration Index, Software Life Cycle Configuration Index, Accomplishment Summary, and any other documents not previously reviewed.

When OOT is used in a project, the Activities/Questions shown in Table 4 below should be considered, **in addition to those in the Job Aid for SOI 4.**

Item #	Evaluation Activity/Question	OOTiA Ref	DO-178B Obj
4.1	Have OO specific aspects been addressed in the Software Accomplishment Summary?	N/A	A10: 3
4.2	Have any deviations been addressed in the Software Accomplishment Summary?	N/A	A10: 3
4.3	Do open problem reports indicate any safety problems or overarching OO problems?	N/A	A10: 3
4.4	Have all configuration items and tools been addressed in the Software Configuration Index and/or Software Life Cycle Environment Configuration Index?	N/A	A10: 3
4.5	Have all items from previous reviews been addressed?	N/A	A10: 3
4.6	Have all issues in the OO Handbook been assessed and properly addressed?	Vol 1-3	ALL

Table 4.3-4 SOI 4 Activities/Questions

4.4 References

1. FAA Order 8110.49, *Software Approval Guidelines*, June 3, 2003. A copy of this Order is available from the FAA website at <http://www2.faa.gov/certification/aircraft/av-info/software/software.htm>.
2. FAA Job Aid, *Conducting Software Reviews Prior to Certification*, dated January 16, 2004. A copy of this FAA Job Aid is available from the FAA website at <http://www2.faa.gov/certification/aircraft/av-info/software/software.htm>.