

RTCA, Inc.
1150 18th Street, NW, Suite 910
Washington, DC 20036

**Object-Oriented Technology and Related
Techniques
Supplement to DO-178C and DO-278A**

Copies of this document may be obtained from

RTCA, Inc.

Telephone: 202-833-9339

Facsimile: 202-833-9434

Internet: www.rtca.org

Please visit the RTCA Online Store for document pricing and ordering information.

FOREWORD

This document was prepared by RTCA Special Committee 205 (SC-205) and EUROCAE Working Group 71 (WG-71). It was approved by the RTCA Program Management Committee (PMC) on December 13, 2011.

RTCA, Incorporated is a not-for-profit organization formed to advance the art and science of aviation and aviation electronic systems for the benefit of the public. The organization functions as a Federal Advisory Committee and develops consensus-based recommendations on contemporary aviation issues. RTCA's objectives include but are not limited to:

- coalescing aviation system user and provider technical requirements in a manner that helps government and industry meet their mutual objectives and responsibilities;
- analyzing and recommending solutions to the system technical issues that aviation faces as it continues to pursue increased safety, system capacity, and efficiency;
- developing consensus on the application of pertinent technology to fulfill user and provider requirements, including development of minimum operational performance standards for electronic systems and equipment that support aviation; and
- assisting in developing the appropriate technical material upon which positions for the International Civil Aviation Organization and the International Telecommunication Union and other interested international organizations can be based.

The organization's recommendations are often used as the basis for government and private sector decisions as well as the foundation for many Federal Aviation Administration Technical Standard Orders.

Since the RTCA is not an official agency of the United States Government, its recommendations may not be regarded as statements of official government policy unless so enunciated by the U.S. government organization or agency having statutory jurisdiction over any matters to which the recommendations relate.

This Page Intentionally Left Blank

TABLE OF CONTENTS

OO.1.0	INTRODUCTION	1
OO.1.1	Purpose	1
OO.1.2	Scope	1
OO.1.3	Relationship to Other Documents	2
OO.1.4	How to Use This Supplement.....	2
OO.1.5	Document Overview	2
OO.1.6	Characteristics of Object-oriented Technology and Related Techniques	3
OO.1.6.1	Basic Concepts.....	3
OO.1.6.1.1	Classes and Objects.....	3
OO.1.6.1.2	Types and Type Safety.....	4
OO.1.6.1.3	Hierarchical Encapsulation	5
OO.1.6.1.4	Polymorphism	5
OO.1.6.1.5	Function Passing and Closures	6
OO.1.6.1.6	Method Dispatch.....	7
OO.1.6.2	Key Features and Related Techniques.....	7
OO.1.6.2.1	Inheritance.....	8
OO.1.6.2.2	Parametric Polymorphism.....	8
OO.1.6.2.3	Overloading (ad hoc polymorphism)	8
OO.1.6.2.4	Type Conversion.....	8
OO.1.6.2.5	Software Exceptions and Exception Handling.....	9
OO.1.6.2.6	Dynamic Memory Management	9
OO.1.6.2.7	Virtualization Techniques.....	10
OO.2.0	SYSTEM ASPECTS RELATING TO SOFTWARE DEVELOPMENT	11
OO.3.0	SOFTWARE LIFE CYCLE	13
OO.4.0	SOFTWARE PLANNING PROCESS	15
OO.4.1	Software Planning Process Objectives	15
OO.4.2	Software Planning Process Activities.....	16
OO.4.3	Software Plans.....	17
OO.4.4	Software Life Cycle Environment Planning	18
OO.4.4.1	Software Development Environment.....	18
OO.4.4.2	Language and Compiler Considerations.....	19
OO.4.4.3	Software Test Environment	19
OO.4.5	Software Development Standards	19
OO.4.6	Review of the Software Planning Process.....	20
OO.5.0	SOFTWARE DEVELOPMENT PROCESSES	21
OO.5.1	Software Requirements Process	21
OO.5.1.1	Software Requirements Process Objectives.....	22
OO.5.1.2	Software Requirements Process Activities	22
OO.5.2	Software Design Process.....	22
OO.5.2.1	Software Design Process Objectives	22
OO.5.2.2	Software Design Process Activities	22
OO.5.2.3	Designing for User-Modifiable Software	23
OO.5.2.4	Designing for Deactivated Code.....	23
OO.5.3	Software Coding Process	23

OO.5.3.1	Software Coding Process Objectives	23
OO.5.3.2	Software Coding Process Activities.....	23
OO.5.4	Integration Process	23
OO.5.4.1	Integration Process Objectives.....	24
OO.5.4.2	Integration Process Activities	24
OO.5.5	Software Development Process Traceability	24
OO.6.0	SOFTWARE VERIFICATION PROCESS.....	25
OO.6.1	Purpose of Software Verification	25
OO.6.2	Overview of Software Verification Process Activities	25
OO.6.3	Software Reviews and Analyses	26
OO.6.3.1	Reviews and Analyses of High-Level Requirements	27
OO.6.3.2	Reviews and Analyses of Low-Level Requirements	27
OO.6.3.3	Reviews and Analyses of Software Architecture.....	27
OO.6.3.4	Reviews and Analyses of Source Code	27
OO.6.3.5	Reviews and Analyses of the Outputs of the Integration Process.....	28
OO.6.4	Software Testing	28
OO.6.4.1	Test Environment.....	28
OO.6.4.2	Requirements-Based Test Selection	28
OO.6.4.2.1	Normal Range Test Cases	29
OO.6.4.2.2	Robustness Test Cases	29
OO.6.4.3	Requirements-Based Testing Methods	29
OO.6.4.4	Test Coverage Analysis	29
OO.6.4.4.1	Requirements-Based Test Coverage Analysis.....	29
OO.6.4.4.2	Structural Coverage Analysis.....	29
OO.6.4.4.3	Structural Coverage Analysis Resolution.....	29
OO.6.4.5	Reviews and Analyses of Test Cases, Procedures, and Results.....	29
OO.6.5	Software Verification Process Traceability.....	30
OO.6.6	Verification of Parameter Data Items.....	30
OO.6.7	Local Type Consistency Verification.....	30
OO.6.7.1	Local Type Consistency Verification Objective.....	30
OO.6.7.2	Local Type Consistency Verification Activity	30
OO.6.8	Dynamic Memory Management Verification	30
OO.6.8.1	Dynamic Memory Management Verification Objective	30
OO.6.8.2	Dynamic Memory Management Verification Activities	30
OO.7.0	SOFTWARE CONFIGURATION MANAGEMENT PROCESS	33
OO.8.0	SOFTWARE QUALITY ASSURANCE PROCESS	35
OO.8.1	Software Quality Assurance Process Objectives.....	35
OO.8.2	Software Quality Assurance Process Activities	35
OO.8.3	Software Conformity Review.....	36
OO.9.0	CERTIFICATION LIAISON PROCESS	37
OO.9.1	Means of Compliance and Planning.....	37
OO.9.2	Compliance Substantiation.....	37
OO.9.3	Minimum Software Life Cycle Data Submitted to Certification Authority.....	38
OO.9.4	Software Life Cycle Data Related to Type Design	38

OO.10.0	OVERVIEW OF CERTIFICATION PROCESS.....	39
OO.11.0	SOFTWARE LIFE CYCLE DATA	41
OO.11.1	Plan for Software Aspects of Certification	41
OO.11.2	Software Development Plan	42
OO.11.3	Software Verification Plan.....	42
OO.11.4	Software Configuration Management Plan.....	42
OO.11.5	Software Quality Assurance Plan	42
OO.11.6	Software Requirements Standards	42
OO.11.7	Software Design Standards	42
OO.11.8	Software Code Standards.....	43
OO.11.9	Software Requirements Data	44
OO.11.10	Design Description.....	44
OO.11.11	Source Code.....	44
OO.11.12	Executable Object Code.....	44
OO.11.13	Software Verification Cases and Procedures	44
OO.11.14	Software Verification Results	44
OO.11.15	Software Life Cycle Environment Configuration Index.....	44
OO.11.16	Software Configuration Index.....	44
OO.11.17	Problem Reports	44
OO.11.18	Software Configuration Management Records.....	44
OO.11.19	Software Quality Assurance Records	45
OO.11.20	Software Accomplishment Summary	45
OO.11.21	Trace Data.....	46
OO.11.22	Parameter Data Item File	46
OO.12.0	ADDITIONAL CONSIDERATIONS	47
OO.12.1	Use of Previously Developed Software	47
OO.12.1.1	Modifications to Previously Developed Software	47
OO.12.1.2	Change of Aircraft Installation	47
OO.12.1.3	Change of Application or Development Environment	47
OO.12.1.4	Upgrading a Development Baseline	47
OO.12.1.5	Software Configuration Management Considerations.....	47
OO.12.1.6	Software Quality Assurance Considerations	47
OO.12.2	Tool Qualification	48
OO.12.3	Alternative Methods.....	48
	OO.12.3.4.2 Sufficiency of Accumulated Service History	48
	ANNEX OO.A – PROCESS OBJECTIVES AND OUTPUTS BY SOFTWARE LEVEL IN DO-178C	49
	ANNEX OO.B – ACRONYMS AND GLOSSARY OF TERMS	61
	ANNEX OO.C – PROCESS OBJECTIVES AND OUTPUTS BY ASSURANCE LEVEL IN DO-278A67	
	ANNEX OO.D –VULNERABILITY ANALYSIS	79
OO.D.1	Key Features and Related Techniques	79
OO.D.1.1	Inheritance.....	79
OO.D.1.1.1	Vulnerabilities.....	79
OO.D.1.1.2	Related Guidance.....	80
OO.D.1.1.3	Supporting Information for Activities	80
OO.D.1.2	Parametric Polymorphism.....	81
OO.D.1.2.1	Vulnerabilities.....	81
OO.D.1.2.2	Related Guidance.....	81

OO.D.1.2.3	Supporting Information for Activities	81
OO.D.1.3	Overloading.....	81
OO.D.1.3.1	Vulnerabilities.....	81
OO.D.1.3.2	Related Guidance.....	82
OO.D.1.3.3	Supporting Information for Activities	82
OO.D.1.4	Type Conversion	82
OO.D.1.4.1	Vulnerabilities.....	82
OO.D.1.4.2	Related Guidance.....	82
OO.D.1.4.3	Supporting Information for Activities	82
OO.D.1.5	Exception Management.....	83
OO.D.1.5.1	Vulnerabilities.....	83
OO.D.1.5.2	Related Guidance.....	83
OO.D.1.5.3	Supporting Information for Activities	83
OO.D.1.6	Dynamic Memory Management (DMM).....	83
OO.D.1.6.1	Vulnerabilities.....	83
OO.D.1.6.2	Related Guidance.....	84
OO.D.1.6.3	Supporting Information for Activities	84
OO.D.1.7	Virtualization.....	86
OO.D.1.7.1	Vulnerabilities.....	86
OO.D.1.7.2	Related Guidance.....	86
OO.D.1.7.3	Supporting Information for Activities	86
OO.D.2	General Issues	87
OO.D.2.1	Traceability	87
OO.D.2.1.1	OOT&RT Specific Considerations	87
OO.D.2.1.2	Related Guidance.....	87
OO.D.2.1.3	Supporting Information for Activities	87
OO.D.2.2	Structural Coverage.....	88
OO.D.2.2.1	OOT&RT Specific Considerations	88
OO.D.2.2.2	Related Guidance.....	89
OO.D.2.2.3	Supporting Information for Activities	89
OO.D.2.3	Component-Based Development.....	90
OO.D.2.3.1	Vulnerabilities.....	90
OO.D.2.3.2	Related Guidance.....	91
OO.D.2.3.3	Supporting Information for Activities	91
OO.D.2.4	Resource Analysis.....	93
OO.D.2.4.1	Annotations and Static Analysis	93
OO.D.2.4.2	Memory Usage Analysis.....	94
OO.D.2.4.3	Timing Analysis.....	99

APPENDIX OO.A – COMMITTEE MEMBERSHIP	A-1
--	-----

APPENDIX OO.B –FREQUENTLY ASKED QUESTIONS	B-1
---	-----

OO.B.1	General Questions	B-1
FAQ #1:	What is covered by this supplement?	B-1
FAQ #2:	What is the relationship between this supplement and the OOTiA Handbook?.....	B-2
FAQ #3:	What additional information is required for the software planning process in an OO related system?.....	B-2
FAQ #4:	What additional measures may be taken when reusing OO components?.....	B-2
FAQ #5:	What additional objectives and activities have been added to support OOT&RT in the software development processes?	B-3

FAQ #6: What additional objectives may be met and activities may be performed when reviewing and analyzing software using OO components?	B-3
FAQ #7: What are the vulnerabilities required to be addressed in section OO.11.1?.....	B-4
FAQ #8: Are there any special concerns for using closures?	B-4
OO.B.2 Requirements Considerations.....	B-5
FAQ #9: What is the relationship between low-level requirements and classes?.....	B-5
FAQ #10: What role do classes play in the design process?	B-5
FAQ #11: When should a requirement be expressed as invariants as opposed to preconditions and postconditions?	B-6
FAQ #12: Do all methods, including constructor, accessor, mutator, and destructor, need requirements?	B-6
OO.B.3 Design Considerations.....	B-6
FAQ #13: What is typing?.....	B-6
FAQ #14: Why is the Liskov Substitution Principle (LSP) important?	B-7
FAQ #15: Must all classes in an application be type consistent?	B-8
FAQ #16: What impact does the Liskov Substitution Principle (LSP) have on architecture and low-level requirements?	B-8
FAQ #17: Are there garbage collection techniques that should be avoided? ...	B-9
FAQ #18: Should inheritance be used for code sharing?	B-9
FAQ #19: What techniques can be used to minimize potential problems with the use of multiple inheritance?	B-9
FAQ #20: Is static dispatch safer than dynamic dispatch?	B-10
FAQ #21: How can Liskov Substitution Principle (LSP) be violated in a strongly-typed language, such as Ada or Java?	B-11
FAQ #22: What would need to be done to make the examples in FAQ #21 conform to the Liskov Substitution Principle (LSP)?.....	B-15
FAQ #23: Does the discussion on preconditions, postconditions, and invariants imply the use of a particular design methodology?	B-15
OO.B.4 Programming Language Considerations	B-15
FAQ #24: What is an example of static dispatch with method overriding? ...	B-15
FAQ #25: Is it acceptable to use a COTS library, such as the C++ Standard Template Library, without explicitly specifying high-level and low-level software requirements, tests, and documents for this library?B-17	B-17
FAQ #26: What considerations are important when using garbage collection?B-17	B-17
FAQ #27: What considerations are important when using a virtual machine?B-18	B-18
OO.B.5 Verification Considerations.....	B-19
FAQ #28: What additional objectives and activities have been added to support OOT&RT in the verification process?.....	B-19
FAQ #29: In section OO.5.5.d, what is meant by “a requirement which traces to a method”?	B-19
FAQ #30: How does the formal verification activity for ensuring local type consistency differ from a low-level requirements review?	B-20
FAQ #31: Does the guidance related to dynamic dispatch also apply to the use of function pointers (for example, function pointer tables and callback functions)?	B-20
FAQ #32: What is the difference between fulfilling type consistency locally and globally?.....	B-21
FAQ # 33: How can local type consistency be verified?.....	B-21
FAQ # 34: How can local type consistency with an abstract superclass be verified by testing without using pessimistic testing?.....	B-21

FAQ #35: How can classes that include unencapsulated data be verified for local type consistency?	B-22
FAQ #36: Does this supplement require tracing of low-level requirements to preconditions, postconditions, and invariants?	B-23
FAQ #37: Is "flattened class testing" a way of verifying local type consistency?	B-23
FAQ #38: When preconditions, post conditions, and invariants are checked at run time, what verification must be performed on the additional Source Code?	B-24
FAQ #39: Is there a problem with covariant collections?	B-24

LIST OF TABLES

TABLE OO.A-1 SOFTWARE PLANNING PROCESS.....	50
TABLE OO.A-2 SOFTWARE DEVELOPMENT PROCESSES.....	51
TABLE OO.A-3 VERIFICATION OF OUTPUTS OF SOFTWARE REQUIREMENTS PROCESS	53
TABLE OO.A-4 VERIFICATION OF OUTPUTS OF SOFTWARE DESIGN PROCESS	54
TABLE OO.A-5 VERIFICATION OF OUTPUTS OF SOFTWARE CODING & INTEGRATION PROCESSES.....	55
TABLE OO.A-6 TESTING OF OUTPUTS OF INTEGRATION PROCESS.....	56
TABLE OO.A-7 VERIFICATION OF VERIFICATION PROCESS RESULTS	57
TABLE OO.A-8 SOFTWARE CONFIGURATION MANAGEMENT PROCESS	58
TABLE OO.A-9 SOFTWARE QUALITY ASSURANCE PROCESS	59
TABLE OO.A-10 CERTIFICATION LIAISON PROCESS	60
TABLE OO.C-1 SOFTWARE PLANNING PROCESS.....	68
TABLE OO.C-2 SOFTWARE DEVELOPMENT PROCESSES	69
TABLE OO.C-3 VERIFICATION OF OUTPUTS OF SOFTWARE REQUIREMENTS PROCESS	71
TABLE OO.C-4 VERIFICATION OF OUTPUTS OF SOFTWARE DESIGN PROCESS	72
TABLE OO.C-5 VERIFICATION OF OUTPUTS OF SOFTWARE CODING & INTEGRATION PROCESSES.....	73
TABLE OO.C-6 TESTING OF OUTPUTS OF INTEGRATION PROCESS.....	74
TABLE OO.C-7 VERIFICATION OF VERIFICATION PROCESS RESULTS	75
TABLE OO.C-8 SOFTWARE CONFIGURATION MANAGEMENT PROCESS.....	76
TABLE OO.C-9 SOFTWARE QUALITY ASSURANCE PROCESS.....	77
TABLE OO.C-10 CERTIFICATION LIAISON PROCESS.....	78
TABLE OO.D.1.6.3 WHERE ACTIVITIES ARE ADDRESSED FOR DMM.....	85

This Page Intentionally Left Blank

OO.1.0 INTRODUCTION

Object-oriented technology (OOT) has been widely adopted in non-critical software development projects. The use of this technology for critical software applications in avionics has increased, but there are a number of issues that need to be considered to ensure the safety and integrity goals are met. These issues are both directly related to language features and to complications encountered with meeting well-established safety objectives. In fact, there are a number of related techniques that are used with OOT that need to be considered as well. Clarifying each of these issues will ease the application of object-oriented technology and related techniques (OOT&RT); therefore, this supplement addresses both object-oriented technology and related techniques.

OO.1.1 Purpose

The purpose of this supplement is to provide guidance for the production of software using OOT&RT for systems and equipment that performs its intended function with a level of confidence in safety that complies with airworthiness requirements. This supplement contains modifications and additions to DO-178C objectives, activities, explanatory text, and software life cycle data that should be addressed when OOT&RT are used as part of the software life cycle.

Compliance with the objectives of DO-178C, Software Considerations in Airborne Systems and Equipment Certification, is the primary means of obtaining approval of software used in civil aviation products. Object-oriented technology is a software development methodology that uses objects and the connections between those objects to express the software design. There are a number of related techniques commonly associated with, but not limited to, object-oriented technology that are considered in this supplement. These related techniques include parametric polymorphism, overloading, type conversion, exception management, dynamic memory management, and virtualization. Since OOT&RT differs from the traditional functional approach to software development, satisfying some of the DO-178C objectives when using OOT&RT may be unclear and/or complicated.

This supplement identifies the additions, modifications, and deletions to DO-178C objectives when object-oriented technology or related techniques are used as part of the software development life cycle and additional guidance is required. This supplement, in conjunction with DO-178C, is intended to provide a common framework for the evaluation and acceptance of OOT&RT based systems.

This supplement provides the following data:

- Objectives for OOT&RT software life cycle processes.
- Descriptions of activities and design considerations for achieving those objectives.
- Descriptions of the evidence that indicate that the objectives have been satisfied.

OO.1.2 Scope

This supplement discusses the use of OOT&RT in the software life cycle for software that is produced in accordance with DO-178C. If the applicant is planning to use OOT or a related technique, then the applicant should comply with the relevant parts of this supplement. The related techniques described in section OO.1.6.2 for which this