

DISSERTATION

ENHANCING FLIGHT TESTING LEVERAGING SOFTWARE TESTING TECHNIQUES  
IMPLEMENTED IN MODEL-BASED SYSTEMS ENGINEERING

Submitted by

Jose L. Alvarado Jr.

Department of Systems Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Fall 2024

Doctoral Committee:

Advisor: Thomas H. Bradley

Daniel Herber  
Steven Simske  
Bret Windom

Copyright by Jose L. Alvarado Jr. 2024

All Rights Reserved



## ABSTRACT

### ENHANCING FLIGHT TESTING LEVERAGING SOFTWARE TESTING TECHNIQUES IMPLEMENTED IN MODEL-BASED SYSTEMS ENGINEERING

The Department of Defense (DoD) is significantly shifting toward digital engineering across all systems engineering lifecycle phases. A vital aspect of this transformation is the adoption of model-based testing methodologies within the Test and Evaluation (T&E) processes. This dissertation investigates a grey box Model-Driven Test Design (MDTD) approach that leverages model-based systems engineering (MBSE) artifacts to create flight test scenarios and plans and compares this novel approach to the traditional document-centric methods. The study utilizes the Systems Modeling Language (SysML) to represent artifacts, enabling a comparative analysis between traditional and MDTD processes. Through a case study involving a training system used by the Air Force Operational Test and Evaluation Center (AFOTEC), the dissertation evaluates the MDTD process's effectiveness in generating validated test scenarios and plans that align with established methods. Two additional case studies demonstrate the reuse of SysML elements across different systems under test (SUT), highlighting the benefits, costs, and practical applications of this approach in operational flight testing. The findings include metrics such as Model Reuse Percentage (MR%), Reuse Value Added (RVA), and System Usability Survey (SUS) scores, which measure the reusability of model artifacts and the usability and effectiveness of the “AFOTEC Methodology” model approach in generating flight test plans. This research underscores the importance of model-based testing in operational flight testing and supports the DoD T&E community's ongoing move toward a fully integrated digital engineering ecosystem.

## ACKNOWLEDGEMENTS

This research effort was made possible through the Defense Acquisition Workforce Development Fund (DAWDF) program administered by the Department of the Air Force (DAF) civilian Scientist and Engineer (S&E) career field funding from the Department of the Air Force (DAF), a division of the Department of Defense (DoD), in support of the Air Force Operational Test and Evaluation Center (AFOTEC), Detachment 5, located at Edwards AFB, California. Furthermore, this funding allows approved government civilians selected through a competitive process to utilize the Science, Technology, Engineering, Mathematics, plus Management (STEM+M) program maintained by the Air Force Civilian Center (AFPC) Career Field team located at Randolph AFB, Texas.

I give my most profound appreciation to those who have contributed ideas, thoughts, and critical comments, without which this work would not have been completed. I want to thank my advisor, Dr. Thomas Bradley, for his guidance, help, and patience throughout this dissertation, where his guidance on the critical components of the process provided clarity and definition that was very much needed throughout the process. I also want to thank the other members of my committee for their time, insights, and recommendations throughout this effort are also greatly appreciated.

Thank you to my colleague and fellow faculty member, Dr. J. Shelley, from California State University's Long Beach Antelope Valley Engineering Program in Lancaster, California, who volunteered countless hours of listening and reading multiple revisions to help me define, organize, and consolidate all the information used for this effort. Your dedication to my endeavor is greatly appreciated.

Finally, and most importantly, thank you to my family, especially my wife Gina, whose unwavering support and encouragement helped me achieve one of my lifetime goals. I would not have been successful without their support and understanding during these past four years, as I spent more time working at school than with them. Thank you for understanding my need to write and work on my schoolwork instead of doing other activities and for giving me the time to complete this degree. Your support has made this possible; I love you and cannot thank you enough for your encouragement throughout this journey.

## DEDICATION

I dedicate this dissertation to my family: my wife, Gina, and all my kids, Stephanie, Alyssa, Toby, Cameron, Ethan, and Gabriel. These people are my pride and joy, and they have always encouraged me to follow my dreams.

To my wife, thank you for your love and support. You challenge me to be better every day. To all my kids, may you strive to find whatever makes you happy and pursue it with all your heart. Finally, to my dad, who said, “An education is the best thing you can get because no one can take that away from you!”

## DISCLAIMER

This dissertation was prepared by Jose Alvarado in his personal capacity and as a student attending Colorado State University. The opinions expressed in this article are the author's own and do not reflect the view of the Air Force, the Department of Defense, or the United States government.

## TABLE OF CONTENTS

|                                                          |     |
|----------------------------------------------------------|-----|
| ABSTRACT .....                                           | ii  |
| ACKNOWLEDGEMENTS.....                                    | iii |
| DEDICATION.....                                          | v   |
| DISCLAIMER.....                                          | vi  |
| LIST OF TABLES .....                                     | xii |
| LIST OF FIGURES .....                                    | xiv |
| Chapter 1 - Introduction .....                           | 1   |
| 1.1 Context.....                                         | 1   |
| 1.2 Funding and Support.....                             | 2   |
| 1.3 Limitations and Constraints .....                    | 2   |
| 1.4 Structure of this Dissertation.....                  | 4   |
| Chapter 2 – Background.....                              | 5   |
| 2.1 Motivation .....                                     | 5   |
| 2.2 DoD Digital Engineering Strategy .....               | 8   |
| 2.3 Flight-Testing Process .....                         | 11  |
| 2.4 Model-Based Systems Engineering (MBSE) Approach..... | 14  |
| 2.4.1 General Concepts of a System .....                 | 14  |
| 2.4.2 Hierarchy within a System .....                    | 15  |
| 2.4.3 Definition of Systems of Systems.....              | 16  |
| 2.4.4 MBSE Overview .....                                | 19  |
| Chapter 3 – Research Questions and Tasks.....            | 22  |

|                                                                |    |
|----------------------------------------------------------------|----|
| 3.1 Overview.....                                              | 22 |
| 3.2 Problem Statement .....                                    | 22 |
| 3.3 Research Focus.....                                        | 23 |
| 3.4 Research Questions and Task List .....                     | 23 |
| 3.4.1 Research Question 1 .....                                | 23 |
| 3.4.2 Research Question 2.....                                 | 26 |
| 3.4.3 Research Question 3.....                                 | 27 |
| Chapter 4 – Literature Review.....                             | 29 |
| 4.1 Introduction.....                                          | 29 |
| 4.2 Model-Driven Test Approach.....                            | 30 |
| 4.3 Software Testing Methodologies .....                       | 34 |
| 4.3.1 Static Analysis and Dynamic Testing .....                | 35 |
| 4.3.2 White Box Testing .....                                  | 36 |
| 4.3.3 Black Box Testing.....                                   | 38 |
| 4.3.4 Grey Box Testing.....                                    | 40 |
| 4.3.5 Nonfunctional Testing Techniques .....                   | 42 |
| 4.3.6 Summary.....                                             | 43 |
| Chapter 5 – Developing Model-Based Flight Test Scenarios ..... | 44 |
| 5.1 Introduction.....                                          | 44 |
| 5.2 Preliminary Work.....                                      | 44 |
| 5.2.1 Research Question 1 Task 1 .....                         | 45 |
| 5.3 Publication Details .....                                  | 47 |
| 5.3.1 Abstract.....                                            | 48 |

|                                                                                     |    |
|-------------------------------------------------------------------------------------|----|
| 5.3.2 Introduction.....                                                             | 49 |
| 5.3.3 Motivation .....                                                              | 50 |
| 5.3.4 Background.....                                                               | 51 |
| 5.3.5 The Flight Test Application.....                                              | 53 |
| 5.3.6 Methods .....                                                                 | 54 |
| 5.3.7 Existing Test Scenario Generation Process .....                               | 56 |
| 5.3.8 Case Study Using Grey Box Test Scenario Generation Process .....              | 59 |
| 5.3.9 Results and Discussion .....                                                  | 67 |
| 5.3.10 Conclusions.....                                                             | 70 |
| 5.4 Remaining Tasks Associated with Research Question 1.....                        | 71 |
| 5.4.1 Software Testing Metrics .....                                                | 72 |
| 5.4.2 Metrics Suitable for Transfer to Flight Test .....                            | 73 |
| 5.5 Addressing Research Question 1 .....                                            | 74 |
| Chapter 6 – A Case Study-Based Assessment of a Model-Driven Testing Methodology and |    |
| Metrics of Model Reuse .....                                                        | 75 |
| 6.1 Introduction.....                                                               | 75 |
| 6.2 Publication Details .....                                                       | 75 |
| 6.2.1 Abstract.....                                                                 | 76 |
| 6.2.2 Introduction.....                                                             | 76 |
| 6.2.3 Background.....                                                               | 78 |
| 6.2.4 Methods .....                                                                 | 80 |
| 6.2.5 Case Studies Definition.....                                                  | 88 |
| 6.2.6 Applicability Assessment Methods.....                                         | 91 |



|                                                                                                                  |     |
|------------------------------------------------------------------------------------------------------------------|-----|
| 6.2.7 Results .....                                                                                              | 98  |
| 6.2.8 Discussion.....                                                                                            | 100 |
| 6.2.9 Conclusion .....                                                                                           | 105 |
| 6.3 Addressing Research Question 2 .....                                                                         | 105 |
| Chapter 7 – Costs And Benefits Of Using A Model-Based Approach In An Example Acquisition Planning Scenario ..... | 107 |
| 7.1 Introduction.....                                                                                            | 107 |
| 7.2 AFOTEC Flight Test Approach.....                                                                             | 107 |
| 7.2.1 Test Planning Process.....                                                                                 | 107 |
| 7.2.2 Operational Test & Evaluation Construct .....                                                              | 108 |
| 7.2.3 Test Design .....                                                                                          | 109 |
| 7.3 Publication Details .....                                                                                    | 112 |
| 7.3.1 Abstract.....                                                                                              | 112 |
| 7.3.2 Introduction.....                                                                                          | 113 |
| 7.3.3 Background.....                                                                                            | 115 |
| 7.3.4 Methods .....                                                                                              | 116 |
| 7.3.5 Assessment Method .....                                                                                    | 122 |
| 7.3.6 Results .....                                                                                              | 124 |
| 7.3.7 Discussion.....                                                                                            | 132 |
| 7.3.8 Conclusion .....                                                                                           | 135 |
| 7.4 Addressing Research Question 3 .....                                                                         | 136 |
| Chapter 8 – Conclusions, Contributions, and Future Work .....                                                    | 138 |
| 8.1 Introduction.....                                                                                            | 138 |

|                                                                           |     |
|---------------------------------------------------------------------------|-----|
| 8.2 Conclusions.....                                                      | 138 |
| 8.3 Research Contributions.....                                           | 141 |
| 8.4 Future Work.....                                                      | 143 |
| REFERENCES .....                                                          | 145 |
| APPENDIX A – Usability Survey.....                                        | 150 |
| APPENDIX B – Combat Search and Rescue Locator SysML Model Diagrams.....   | 156 |
| APPENDIX C – E-X Airborne Multi-Sensor Platform SysML Model Diagrams..... | 169 |

## LIST OF TABLES

|                                                                                                                                                                 |     |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Table 1. Borrowed Scientific Community Vocabulary Used in Systems Engineering .....                                                                             | 15  |
| Table 2. Behaviors that are Defined Within the Community of Software Engineering .....                                                                          | 15  |
| Table 3. N-by-N Matrix of Software Testing Vocabulary and Flight-Testing Vocabulary.....                                                                        | 45  |
| Table 4. N-by-N Matrix of Software Techniques .....                                                                                                             | 46  |
| Table 5. All Configurations of Flows are Arranged into Various Scenarios .....                                                                                  | 65  |
| Table 6. Scenario 1 Comparison of the Document-Centric DTP Steps with the Model-Centric<br>Process-Generated Steps Derived from the SysML Activity Diagram..... | 68  |
| Table 7. Summary Table of Case Study Model Characteristics .....                                                                                                | 91  |
| Table 8. Name of Elements Contained Within the CSARL Model Structure Subfolders. ....                                                                           | 94  |
| Table 9. Name of Elements Contained Within the E-X Model Structure Subfolders. ....                                                                             | 95  |
| Table 10. Name of Elements Contained Within the NGS Model Structure Subfolders.....                                                                             | 96  |
| Table 11. Summary Table of AFOTEC Methodology Model Elements for the Two Case Study<br>Models and the Baseline Case Model .....                                 | 97  |
| Table 12. Reuse of Model Elements Between the Baseline CSARL and the E-X Platform Case<br>Study Models .....                                                    | 101 |
| Table 13. Reuse of Model Elements Between the E-X Platform and the NGS System Case Study<br>Models .....                                                        | 102 |
| Table 14. Determination of Experience Level Based on Participant's Self-Evaluation. ....                                                                        | 123 |
| Table 15. Classification of SUS Scores Based on Adjective Rating. ....                                                                                          | 124 |
| Table 16. Distribution of Participant SUS Scores Based on Adjective Rating and Participant<br>Experience Level. ....                                            | 125 |

|                                                                                                          |     |
|----------------------------------------------------------------------------------------------------------|-----|
| Table 17. Stratification of SUS Scores Based on Acceptability Rating.....                                | 126 |
| Table 18. Distribution of Participant SUS Scores Based on Acceptability Rating and Experience Level..... | 126 |
| Table 19. Effectiveness Data Collected During the Survey and Sorted Based on Responses. ...              | 127 |
| Table 20. Value Data Collected During the Survey and Ordered Based on Responses. ....                    | 128 |
| Table 21. Quality Data Collected During the Survey and Order Based on Responses. ....                    | 129 |
| Table 22. Reduction of Time Achieved by using the CAG Draft Test Plan.....                               | 131 |
| Table 23. Reduction Time Achieved by using the CAG Draft Test Plan Decomposed by Experience Level. ....  | 132 |
| Table 24. Opinion on Recommending a Test Team for Use in Writing a Final Test Plan. ....                 | 132 |

## LIST OF FIGURES

|                                                                                                                                                                         |    |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| Figure 1. Major Capability Acquisition Process.....                                                                                                                     | 7  |
| Figure 2. Five Elements of the DoD Defense Digital Engineering Strategy.....                                                                                            | 10 |
| Figure 3. A Statistical Measure of the Verification of a Capability or Performance.....                                                                                 | 13 |
| Figure 4. Concept of a System Hierarchy .....                                                                                                                           | 16 |
| Figure 5. Example of a System of Systems of an Operational Base.....                                                                                                    | 18 |
| Figure 6. Systems Engineering Vee Model.....                                                                                                                            | 21 |
| Figure 7. Model-Driven Test Design .....                                                                                                                                | 32 |
| Figure 8. Activities of Test Engineers.....                                                                                                                             | 33 |
| Figure 9. Hierarchy of Software Testing Methodologies.....                                                                                                              | 35 |
| Figure 10. Model-Driven Test Design (black boxes) Super-Imposed with the Flight Test Planning Process (yellow boxes).....                                               | 56 |
| Figure 11. Grey Box Implementation of the Test Scenario Generation Process.....                                                                                         | 58 |
| Figure 12. Excerpt of Table of Contents from CSARL Detailed Test Plan.....                                                                                              | 60 |
| Figure 13. SysML Use Case Diagram of the Combat Search and Rescue Locator (CSARL) System used in the Case Study. ....                                                   | 61 |
| Figure 14. SysML Activity Diagram of the Combat Search and Rescue Locator (CSARL) System Used in the Case Study with Identified Path Flows.....                         | 62 |
| Figure 15. SysML Sequence Diagram of the Combat Search and Rescue Locator (CSARL) System used in the Case Study. ....                                                   | 63 |
| Figure 16. Activity Diagram of Scenario 1 Illustrating the Test Procedure that Outlines the Steps Required to Execute the SAR Team Circumnavigate to IP Test Case. .... | 66 |

|                                                                                                                                                                         |     |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| Figure 17. This Activity Diagram of Test Scenario 3 Illustrates the Test Procedure, which Outlines the Steps Required to Execute the “Perform Zeroize Procedure.” ..... | 70  |
| Figure 18. Measure Hierarchies and the Overall AFOTEC OT&E Construct.....                                                                                               | 82  |
| Figure 19. MR% and RVA Metrics as Applied Between the Generations of Case Study Models. ....                                                                            | 88  |
| Figure 20. The CSARL Model Layout Incorporates the AFOTEC Test Model, Labeled “AFOTEC Methodology.” .....                                                               | 92  |
| Figure 21. Measure Hierarchies and the Overall AFOTEC OT&E Construct.....                                                                                               | 109 |
| Figure 22. E-X Model Layout Structure Incorporates the AFOTEC Test Model in a Folder Labeled AFOTEC Methodology.....                                                    | 119 |
| Figure 23. A Comparison of the Adjective Ratings, Acceptability Scores, and School Grading Scales in Relation to the SUS Score [67]......                               | 122 |
| Figure 24. Bar graph Illustrating the Distribution of the Participant’s SUS Scores Based Adjective Rating and Experience Level.....                                     | 125 |
| Figure 25. Bar Graph Illustrating the Distribution of the Participant’s SUS Scores Based on Acceptability Rating and Experience Level. ....                             | 126 |
| Figure 26. Box Plot Showing the Distribution of the SUS Scores (n=21).....                                                                                              | 127 |
| Figure 27. Bar graph Illustrating the Results of the Value of the CAG Draft Test Plan. ....                                                                             | 129 |
| Figure 28. Bar graph Illustrating the Results of the CAG Draft Test Plan Quality.....                                                                                   | 131 |
| Figure 29. The Red Line Indicates the Calculated Average Participants’ SUS Score to the Adjective Ratings, Acceptance Ranges, and the SUS score.....                    | 133 |

## CHAPTER 1 - INTRODUCTION

### *1.1 Context*

The Department of Defense (DoD) leadership constantly needs timely information and accurate assessments of “real-world” threats to make crucial decisions. The urgency of this need cannot be overstated. DoD needs weapon systems that can promptly execute these decisions. The Pentagon, home to the five military branches, creates acquisition programs that develop the systems to meet the DoD’s demands for sophisticated multi-role weapons necessary to combat these “real-world” threats. Therefore, DoD leadership requires system performance and demonstrated capabilities to define and assess the parameters needed to make these decisions. The Director of Operational Test and Evaluation (DOT&E) advises DoD leadership by reporting required system performance and demonstrated capabilities for weapons systems currently in the acquisition process.

The DoD acquisition strategy has produced world-leading military systems that have proven capable of executing the mission for which they were designed. Advances in technology and the integration of sophisticated cyber-physical and software-enabled functions have become central to these weapon systems, making them more complicated to design, manufacture, test, and sustain. More sophisticated weapon systems require escalated timelines and costs to build and make them available to the warfighter. The current acquisition process has reached an unsustainable level in which it takes decades to produce modern weapon systems plagued by cost overruns, late schedules, and unsatisfied expectations. “The current acquisition process uses a linear acquisition strategy known as the ‘waterfall’ method that gave military planners the illusion of stability, as firm end-requirements would be determined early in the development process. As

a result, key development decisions would be made before sufficient knowledge was available to make accurate assessments” [1]. This default, serial approach drives long cycle times, makes it challenging to incorporate warfighter experiences and feedback, lacks the flexibility for timely insertion of emerging technologies, and is further plagued by budget instabilities during the long procurement timeline by being subjected to schedule changes, cost increases, and new requirements.

## ***1.2 Funding and Support***

The proposed research aligns with work to support the Air Force Operational Test and Evaluation Center (AFOTEC), Detachment 5, located at Edwards AFB, California, funded by the Defense Acquisition Workforce Development Fund (DAWDF) program. This funding is administered by the Department of the Air Force (DAF) civilian Scientist and Engineer (S&E) career field that allows approved government civilians selected through a competitive process to utilize the Science, Technology, Engineering, Mathematics, plus Management (STEM+M) program maintained by the Air Force Civilian Center (AFPC) Career Field team located at Randolph AFB, Texas, to pursue a second masters or doctoral degree as a full-time assignment for a selected period.

## ***1.3 Limitations and Constraints***

The ANSI/IEEE 1059 standard [2] defines software testing as analyzing a software item to detect the differences between existing and required conditions (that is, defects/errors/bugs) and to evaluate the features of the software item. Software testing has many facets and broad applications that can be used in multiple scenarios. Testing applications in which everything is known about the application is known as “white box” testing. In this area, inputs, outputs, and internal processes can be monitored or manipulated during testing to see how the application



responds to stimuli. Another type of testing is “black box” testing, which is a method of testing an application without knowing the internal structure, design, or implementation. Thus, the only functions that can be observed and measured are the application inputs and related outputs. Then, there is the “grey box,” where partial knowledge of the application’s internal structure is known. These approaches can and are often automated in software applications and run multiple times to create a rich data set from which to compare and derive statistics.

Flight testing, a form of black box testing, presents unique challenges due to the limited insight into the internal structure, design, or implementation of the system under test (SUT). While testers and operators may occasionally gain some insight into the SUT’s design and process flow during initial development, flight tests are primarily categorized as black box or grey box testing. As a result, the techniques, tools, and metrics used in flight testing are constrained to those applicable to black box and grey box testing. Moreover, the nature of flight tests, which aim to control variables, makes it challenging to replicate each experiment consistently.

Adding to the complexity, there are situations in which testing focuses on understanding the behavior or performance of the system under different inputs. Once again, to constrain the problem set, only those techniques, tools, and metrics used in a controlled environment will be considered for this research. This subset of T&E is called “controlled testing” and is defined in two ways: a test can constrain certain variables while allowing others to vary to see what effects the fluctuating variables have on the system, or a test that defines the variables during the initial start of the test and allows them to vary throughout the test to understand how the SUT interacts with other systems or in a given environment.

## ***1.4 Structure of this Dissertation***

This research aims to develop a model-based process for utilizing and extracting documentation and artifacts from a given SUT model. The goal is to design, develop, and generate a flight test plan that can be executed to collect the necessary data for analyzing and evaluating the SUT's performance.

The structure of this dissertation is as follows: Chapter 2 provides a background discussion outlining the DoD acquisition process and the flight test process used to verify and validate the system performance of a SUT. The chapter highlights the DoD's digital engineering strategy with a brief discussion of MBSE as it applies to this dissertation. Chapter 3 reviews the research questions and tasks associated with this dissertation, while Chapter 4 discusses the literature research conducted and used as a basis for this dissertation. Chapter 5 focuses on Research Question 1, which explores the software testing techniques discovered during the literature review that can be implemented in flight tests. The chapter introduces the first MBSE model used as a test case for applying and assessing the software testing techniques identified. Chapter 6 centers on Research Question 2, which introduces two other MBSE models used as test cases to discuss model artifacts and the metrics associated with measuring artifact reusability. Chapter 7 focuses on and presents the survey results used to analyze and assess the usefulness, effectiveness, and usability of generated test plans from information and artifacts contained within the test case models. Finally, Chapter 8 summarizes each subsequent chapter's discussions and research contributions and provides concluding remarks and prospective avenues for further exploration. This dissertation contains works published in journals or presented at conferences. In these cases, the works are reproduced within this dissertation and have been reformatted to meet the dissertation style guidelines. The content has not been altered.

## CHAPTER 2 – BACKGROUND

### *2.1 Motivation*

The Department of Defense (DoD) defines the Test and Evaluation (T&E) process as the planning, conduct, and reporting of a SUT to independently evaluate system performance through developmental and operational T&E with minimal contractor influence. The overarching functions of T&E are to mature system designs, manage risks, identify and help resolve deficiencies as early as possible, and ensure systems are operationally mission capable [3]. T&E is essential in verifying and validating systems using representative articles and ensuring models are well-positioned to meet future production and sustainability needs. Acquisition programs benefit from integrating developmental and operational T&E, including testers, for the acquisition process and the end of the acquisition life cycle. Thus, testers serve not only to support verification, validation, and accreditation but also to provide expertise in building and maturing the models used for various purposes [4].

Threat modeling and simulated combat environments seek to be operationally representative, and verifying and validating the resulting models is an ongoing process throughout system development [4]. Developing a model of the SUT is crucial to designing adequate and appropriate tests of both the model and the system, as well as how the model will be used in the T&E process for the physical system. When coupled with validated data, these models create a digital twin. A Digital Twin is a virtual replica of a process, product, or service. It compares against the physical counterpart for modeling, prognosis, or analysis, thus connecting the physical and digital realms. [5]. T&E feeds into the authoritative source of truth for validating or calibrating digital twins. T&E supports continuous evaluation and decision-making throughout a system's

lifecycle by taking requirements and data from a system's digital twin and testing the models, prototypes, and physical articles to provide performance data to support the verification and validation along with the accreditation of the digital twin [6].

In a typical “model-test-validate” cycle, models of physical systems are used to provide initial estimates of system performance and to inform test plans by identifying what data needs to be collected and how to collect it. Data collected during tests in various environments (such as installed system test facilities or open-air ranges) demonstrate system performance and is fed back into the modeling process to validate or update models. These validated models then form the basis for refining system designs or informing other M&S requirements, such as determining how system performance contributes to the mission and campaign-level effectiveness [6]. In summary, precise validation and verification (V&V) abilities, which effectively quantify capabilities, limitations, and uncertainties, are essential for ensuring the long-term success of the Department of Defense's missions.

The DoD's acquisition process consists of an event-based progression in which a program proceeds through phases, milestones, and reviews. A milestone is the culmination of progressions that are executed during a given phase to determine if a program can proceed to the next segment [7]. Figure 1 illustrates the progression of a significant acquisition system life cycle.

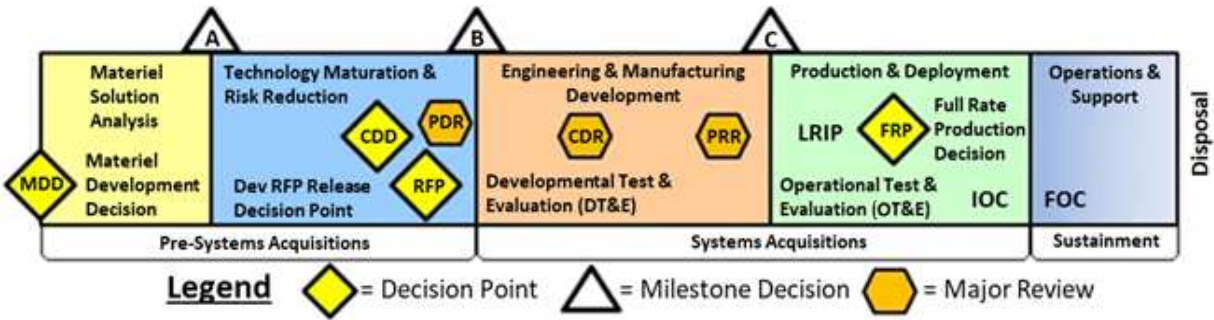


Figure 1. Major Capability Acquisition Process

The structure of DoD acquisition programs is based on strategies that provide oversight that include program information, acquisition phase content, timing and scope of decision reviews, and decision levels that are based on the specifics of the product being acquired, including complexity, risk factors, and required timelines to satisfy validated capability requirements [7]

The focus of this research is on the processes after milestone B. The Engineering and Manufacturing Development (EMD) phase is where a system is developed and designed before production. EMD starts after the successful completion of milestone B and is where system integration, interoperability, supportability, safety, and utility are demonstrated. During EMD, a demonstration of reliability, availability, maintainability, and sustainability is also performed and evaluated in the design of the system [8]. The EMD phase of a program is considered complete once the system capability requirements and the availability of industrial production capabilities have been demonstrated and the manufacturing and software sustainment processes are functional.

Once a system has matured enough to create a prototype, the program will begin Developmental Test and Evaluation (DT&E), where system integration, interoperability, supportability, safety, and utility are tested. DT&E activities include the demonstration that the engineering design and development process is complete, that the design risks have been minimized, and that the system will meet specifications [9]. Program management is informed of the status of hardware and software development progress and product compliance using Key

Performance Parameters (KPPs) and Key System Attributes (KSAs) and how these metrics are being satisfied [8]. At this point, Milestone C is granted, and the next stage of the acquisition process is entered.

The next stage, the Production and Development (PD) Phase, is where the program is allowed to begin low-rate initial production (LRIP) and Operation Test and Evaluation (OT&E) [10]. OT&E's purpose is to test a system in an actual or simulated environment for which it was designed with users of the system in a realistic operating condition to determine operational effectiveness, suitability, and combat survivability [11]. Operational testing is conducted by an organization that is independent of the development agency [11] and reports directly to the Director of Operational Test and Evaluation (DOT&E) of the Department of Defense. For example, the Air Force Operational Test Agency (OTA) is the Air Force Operational Test and Evaluation Center (AFOTEC), which is chartered to conduct operational testing for the United States Air Force (USAF). AFOTEC assesses the capability of new systems to meet warfighter needs by planning, executing, and reporting independent operational evaluations. From concept development through system employment, AFOTEC provides effectiveness, suitability, and operational impact expertise in the battlespace environment [12].

## ***2.2 DoD Digital Engineering Strategy***

According to former Secretary of Defense Donald Rumsfeld, "The DoD (Department of Defense) acquisition process is not getting new systems to the field quickly enough. The current average acquisition response time is ten years, which is longer for major weapon systems. We are delivering technology that is at least a generation old. To reverse this trend, we must change our ways of doing business..." [13]. Furthermore, Dr. Will Roper, the Assistant Secretary of the Air Force for Acquisition, Technology, and Logistics, described the concept of the "digital trinity,"

comprising digital engineering and management, combined with agile software and open architecture, as the next significant paradigm shift for military tech dominance. He detailed that digital design technologies will lead to a faster, more agile, and more competitive weapons-buying process and escape the traditional defense acquisition process [14]. In response to these challenges, the Department of Defense published the Defense Digital Engineering Strategy in June of 2018 to modernize how the Department designs, develops, delivers, operates, and sustains systems.

DoD defines digital engineering as an integrated digital approach that uses authoritative sources of system data and models as a continuum across disciplines to support lifecycle activities from concept through disposal. This approach connects processes, data, and capabilities that will enable the use of models to digitally represent the system of interest throughout its lifecycle and provide the following expected benefits [15]:

- a. Informed decision-making and greater insight through increased transparency.
- b. Enhanced communication.
- c. Increased understanding is intended for greater flexibility and adaptability in design.
- d. Increased confidence that the capability will perform as expected.
- e. Increased efficiency in engineering and acquisition practices.

This strategy, graphically depicted in Figure 2, outlines five elements the department deems essential for a digital engineering ecosystem to succeed [15]:

- a. Formalize the development, integration, and use of models to inform enterprise and program decision-making.
- b. Provide an enduring, authoritative source of truth.
- c. Incorporate technological innovation to improve engineering practice.

- d. Establish a supporting infrastructure and environment to perform activities, collaborate, and communicate with stakeholders.
- e. Transform the culture and workforce to adopt and support digital engineering across the life cycle.



Figure 2. Five Elements of the DoD Defense Digital Engineering Strategy.

The DoD Defense Digital Engineering Strategy’s five elements are what the department deems essential for a digital engineering ecosystem to succeed and how the department develops and sustains systems.

The goal of the first strategy is to establish formal planning, development, and use of models as an integral part of performing engineering activities as a continuum across the lifecycle, while the second strategy’s goal is to move from a document-centric system to digital models with a standard set of data that is current, authoritative, and consistent. The third strategy extends beyond the traditional model-based approaches to incorporate technological improvements and support the speedy implementation of modernizations within the connected digital enterprise. The goal of the fourth strategy is to establish and transform the environment by establishing a robust infrastructure that supports digital engineering goals. The fifth strategy incorporates best practices



of change management and strategic communications to transform the culture and workforce [15]. Thus, the current acquisition process will be converted to a more cooperative integrated information exchange throughout the lifecycle of the SUT.

### ***2.3 Flight-Testing Process***

Flight testing is of interest to this research effort. It comprises two distinct stages. The first stage of flight testing occurs once a program reaches milestone B and a representative prototype is manufactured. DT&E, as stated in the previous section, is where system integration, interoperability, supportability, safety, and utility are tested, and its goal is to demonstrate that the engineering design and development process is complete, the design risks have been minimized, and the system will meet specifications. The second stage of flight testing is when a program reaches the PD Phase and begins LRIP. OT&E's purpose is to determine the operational effectiveness, suitability, and combat survivability of a SUT in an actual or simulated environment for which it was designed to operate with users of the system in a realistic operating condition.

DT&E flight testing is primarily conducted by the Air Force Flight Test Center (AFTC), where comprehensive test plans are assembled and authored "to communicate technical detail and logistics required to execute and report results of flight, ground, and laboratory test of air vehicles, subsystems, and components" [16]. Once these test plans and their associated safety planning are approved, test execution begins, and data is collected for analysis and reporting.

AFOTEC's flight testing is broken up into two sections: Initial Operational Test and Evaluation (IOT&E), which is primarily done when a significant system composed of multiple sub-systems is first developed by the Air Force such as a fighter, bomber, or transport aircraft, and Follow-On Test and Evaluation (FOT&E) which is conducted when a part of a significant system

is either being replaced, upgraded, or discovered deficiencies are corrected. Since FOT&E only focuses on subsystems or parts of a sub-system, regression testing on the system must ensure that functions and capabilities have not been compromised or degraded, making regression testing accountable for existing capabilities' overall stability and functionality. Regression Testing is defined as testing that is accomplished to check that recent changes do not adversely affect existing features. Regression Testing can be executed as a whole, or a partial selection of already executed test cases can be re-executed to ensure existing functionalities work. Both DT&E and OT&E usually conduct FOT&E together to reduce the logistics and assets required for testing and increase data sharing and analysis for quicker reporting of results.

Test planning is conducted no matter which category the proposed testing falls into; thus, many resources are constantly being tasked with accomplishing previous testing to ensure the functionality of the SUT. Since verification of the capability or performance of a SUT is currently being performed using previous test plans, significant and insignificant tests are executed, and the collected data are compared to previous results. Time and budget influence how much regression testing can be accomplished; thus, limited test points and data are collected during recurring regression test execution.

The primary statistical measure used to verify the capability or performance of an SUT is the calculation of confidence intervals based on a given significance level. The current AFTC and AFOTEC decision-making process asks: Does the performance specification fall within the interval? How does the new data set compare with the old data set and its confidence interval? A flow diagram of this process is shown in Figure 3. Analysts must deal with the difference in the sample sizes of the two data sets since accomplishing limited test conditions leads to different sample sizes of test data being gathered between previous and present data sets. A much smaller

sample data set than the old one increases sample uncertainty and the associated confidence interval.

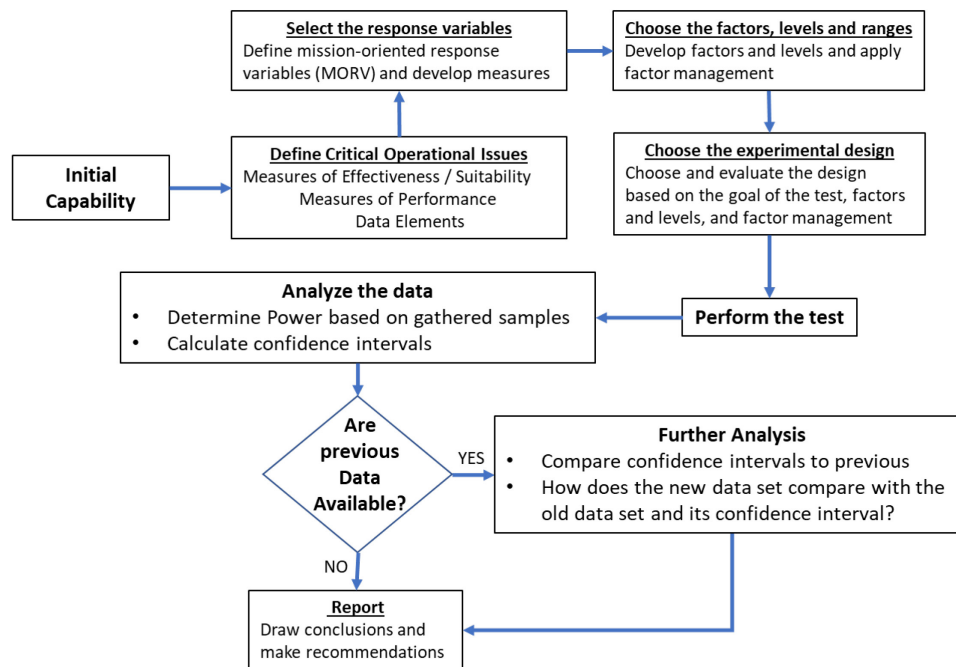


Figure 3. A Statistical Measure of the Verification of a Capability or Performance.

This flow chart describes the process used to determine a capability or performance measurement using the Design of Experiments (DOE) to provide a statistical value to make decisions about an SUT's effectiveness or suitability.

As SUTs evolve and the acquisition community is pressured to field systems quicker, re-performing an entire previous test plan for regression testing will become unfeasible. Thus, techniques found in Design of Experiments (DOE) are being explored to sample the SUT capability and performance envelope to gain a better data set with smaller sample sizes while still getting suitable confidence intervals at high significance levels. But once again, this is a tradeoff between time and cost versus the amount of data needed to answer the questions.

## ***2.4 Model-Based Systems Engineering (MBSE) Approach***

### ***2.4.1 General Concepts of a System***

The International Council on Systems Engineering (INCOSE) describes a system as “... an integrated set of elements, subsystems, or assemblies that accomplish a defined objective. These elements include products (hardware, software, firmware), processes, people, information, techniques, facilities, services, and other support elements” [17]. Furthermore, the concept of the functionality of a “system is typically expressed in terms of the interactions of the system with its operating environment, especially the users,” which leads to the concept of system architecture, which is “defined as the fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution” [17].

Specific vocabulary in systems engineering is summarized in Table 1 [17]. In software engineering, system elements, referred to as software objects, have processes or operations in addition to attributes that have binary logical values that can be either idle or executing; thus, to describe the state of a system, it requires that both attributes and processes be assigned values [17]. Two broad categories of behavior are defined within the software engineering community and are discussed in Table 2 [17]. Furthermore, the concept of the black box/white box system representation is used for problem-solving, which is defined as follows: “Black box representation is based on an external view of the system (attributes).” In contrast, “white box representation is based on an internal view of the system (attributes and structure of the elements)” [17].

It's essential to comprehend the connection between the two concepts, with the system being represented by both its external attributes and its internal attributes and structures [17].

Table 1. Borrowed Scientific Community Vocabulary Used in Systems Engineering

“The repeatability of experiments demanded by science is critical for delivering practical engineering solutions having commercial value. Engineering in general and SE draw heavily from the terminology and concepts of science” [17].

| <b>Terminology</b>            | <b>Definition</b>                                                                                                               |
|-------------------------------|---------------------------------------------------------------------------------------------------------------------------------|
| Attribute (or system element) | Observable characteristic or property of the system (or system element). Attributes are represented symbolically by variables.  |
| Variable                      | Symbol or name that identifies an attribute. Every variable has a domain, which could be but is not necessarily measurable.     |
| Measurement                   | The outcome of a process in which the system of interest (SOI) interacts with an observation system under specified conditions. |
| Value                         | The outcome of a measurement is assigned to a variable.                                                                         |
| State                         | When a system under an assigned value(s) to its attributes remains constant or steady for a meaningful period.                  |

Table 2. Behaviors that are Defined Within the Community of Software Engineering

“A complete description of a system state requires values to be assigned to both attributes and processes” [17].

| <b>Behavior</b> | <b>Definition</b>                                                                                                |
|-----------------|------------------------------------------------------------------------------------------------------------------|
| Dynamic         | The time evolution of the system state.                                                                          |
| Emergent        | The system behavior cannot be understood exclusively in terms of the behavior of the individual system elements. |

#### ***2.4.2 Hierarchy within a System***

The Institute of Electrical and Electronics Engineers (IEEE) defines a system element as either atomic (in which it cannot be further decomposed) or as a system of merit (which can be decomposed into further subordinate system elements) [17]. In either case, “the integration of the system elements must establish the relationship between the effects that organizing the elements has on their interactions and how these effects enable the system to achieve its purpose” [17].

When defining the elements of a system and their interconnections, “... a set of distinct system elements with regard only to their relation to the whole (system) by suppressing details of their interactions and interrelations ... is referred to as a partitioning of the system...”, and “... at any given level, the elements are grouped into distinct subsets of elements subordinated to a higher-level system” [17]. This is illustrated in Figure 4.

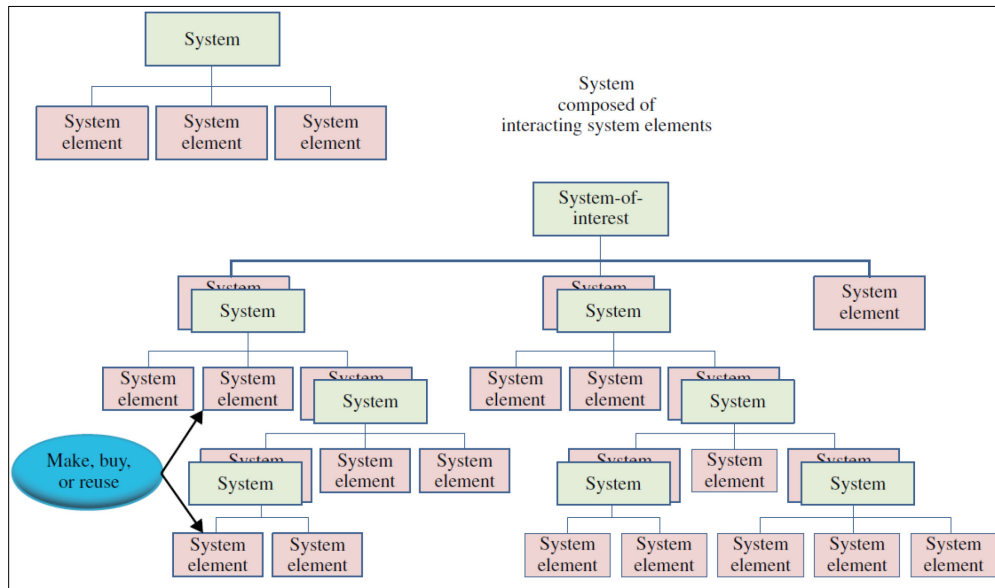


Figure 4. Concept of a System Hierarchy

The concept of a system hierarchy is adapted from ISO/IEC/IEEE 15288:2015, in which the system life cycle processes ... are described as a system composed of a set of interacting system elements, each of which can be implemented to fulfill its specified requirements. [17].

### 2.4.3 Definition of Systems of Systems

A system of systems (SoS), as defined by INCOSE, “is an SOI (system of interest) whose elements are managerially and operationally independent systems. These interoperating and integrated collections of constituent systems usually produce results unachievable by the individual systems alone” [17]. Since an SoS is a system within itself, a choice to address it as

either a system as a whole or as an SoS depends on which perspective is better suited to address the problem, and the following characteristics can be used to decide which tactic is better [17]:

- a. Operational independence of constituent systems
- b. Managerial independence of constituent systems
- c. Geographical distribution
- d. Emergent behavior
- e. Evolutionary development processes

Figure 5 illustrates the concept of an SoS by defining the air transport system as an SoS comprising multiple aircraft, airports, air traffic control systems, and ticketing systems. Other systems, such as security and financial systems, facilitate passenger transportation [17].

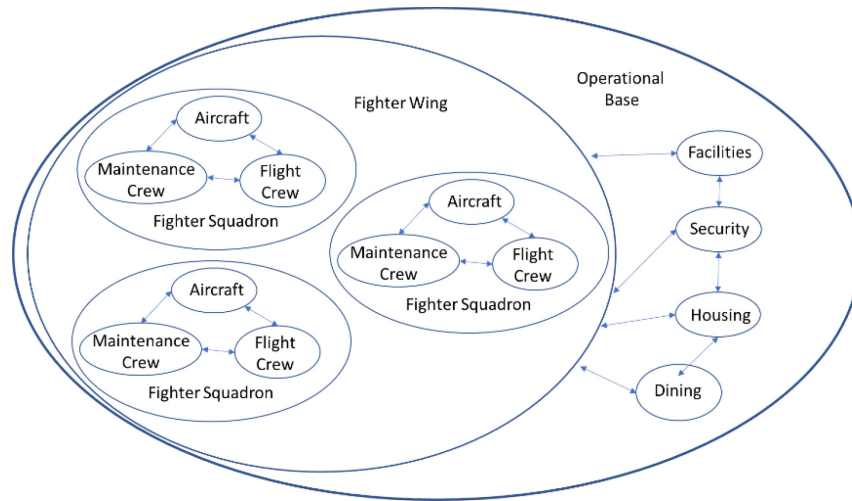


Figure 5. Example of a System of Systems of an Operational Base

An Air Force operational base is an SoS comprising multiple aircraft, aircrews, squadrons, maintenance, base support systems, and security systems that facilitate Airspace surveillance and protection.

Since an SoS is composed of independent component structures, SoS's poses challenges in conducting end-to-end SoS testing, and unless there is an understanding of the SoS-level expectations and measures, it can be problematic to evaluate the level of performance, determining areas that need attention, and confirming the capabilities and limitations of the SoS to the users [17]. Despite having a clear understanding of the objectives and metrics, depending on the SoS context, there may be multiple factors that make it difficult, if not impossible, such as financial limitations to SoS testing, when multiple constituent systems within the SoS development cycles are asynchronous, or when an SoS is large and diverse, making complete end-to-end testing challenging or prohibitively costly [17]. "Often, the only way to get a good measure of SoS performance is from data collected from actual operations or through estimates based on modeling, simulation, and analysis" [17].



#### ***2.4.4 MBSE Overview***

INCOSE states that MBSE is a “formalized application of modeling to support system requirements, design, analysis, verification, and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases” [17]. The MBSE process improves the ability to capture, analyze, share, and manage information associated with the specification of a product and results in the following improvements and enhancements [17]:

- a. Communication among the development stakeholders.
- b. Ability to manage system complexity by permitting a system model to be examined from various viewpoints and analyze the impacts of changes to the system.
- c. Ensure product quality by furnishing a system model that enables assessment for consistency, correctness, and completeness.
- d. Capturing and reusing knowledge involves standardizing information capture and utilizing model-driven methods to streamline the design modification process, ultimately reducing cycle time and cutting maintenance expenses.

With a traditional document-based systems engineering (SE) approach, considerable information is produced about the system, such as specifications, interface control documents (ICDs), system description documents (SDDs), trade studies, analysis reports, and verification plans, procedures, and reports, that contain information that is often difficult to maintain, synchronize, and assess in terms of its quality [17]. With an MBSE approach, since the information is depicted in a system model or set of models, this information is maintained throughout the life cycle and results in significant improvements in system requirements, architecture, and design quality, lowering the risk and costs of system development, enhancing productivity through reuse

of system artifacts, and improving communications among the system development participants [17].

Since the MBSE methodology is a collection of related processes, methods, and tools organized in a “model-based” or “model-driven” context used to support a specific product or SoS, many different MBSE approaches have been proposed. In 2008, an INCOSE technical publication published a survey of MBSE methodologies. Six candidate MBSE methodologies were surveyed: INCOSE Object-Oriented Systems Engineering Method (OOSEM), IBM Rational Telelogic Harmony-SE, IBM Rational Unified Process for Systems Engineering (RUP-SE), Vitech MBSE Methodology, JPL State Analysis (SA), and Dori Object-Process Methodology (OPM) [17]. Of these six approaches, the OOSEM methodology is best suited for regression testing since it “integrates object-oriented concepts with model-based and traditional SE methods to help architect flexible and extensible systems that accommodate evolving technologies and changing requirements” and is consistent with the typical “Vee” SE process shown in Figure 6, and can be applied recursively and iteratively at each level of the system hierarchy [17].

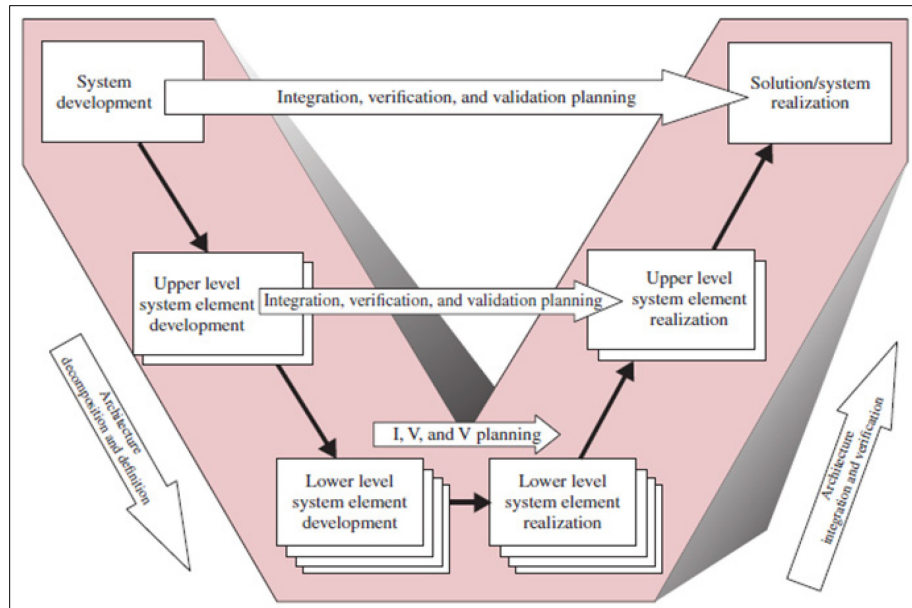


Figure 6. Systems Engineering Vee Model

The Vee model, introduced in 1991, is a sequential method used to visualize various critical areas for SE focus, particularly during the concept and development stages. The Vee highlights the need for continuous validation with the stakeholders, the need to define verification plans during requirements development, and the importance of constant risk and opportunity assessment [17].

## CHAPTER 3 – RESEARCH QUESTIONS AND TASKS

### *3.1 Overview*

The overarching purpose of DoD T&E is to provide information to decision-makers to determine whether to field a weapon system. In this context, finding the right balance in T&E is crucial, enabling swift planning and execution of test phases and events. These activities should generate statistically relevant data, facilitating independent analysis, evaluation, and reporting to fulfill the DoD's objective of swiftly introducing new capabilities.

New Air Force acquisition programs are mandated to follow and use an MBSE approach. Currently, the T&E community must abandon relying solely on traditional document-centric methods and embrace a model-centric approach. This shift is vital for managing program complexities and seamlessly integrating technologies and design strategies to achieve all stakeholders' developmental and operational test objectives within the existing system development resources. Furthermore, test planning personnel need to create and modify test plans as requirements mature to verify specifications and mission objectives that can be statistically defensible to inform senior leaders of system performance and how parameters affect system effectiveness, suitability, and maintainability.

### *3.2 Problem Statement*

This research proposes and develops a methodology implemented in an MBSE construct that provides inputs to the AFOTEC test planning processes. Using systems engineering artifacts, such as mission domain elements, requirements, use cases, and activities, as a trustworthy source for all stakeholders improves the reuse of previous data and test plans, helps the flight test

community improve test point efficiency, and aids in data analysis. This implementation within the MBSE paradigm was analyzed for its ability to improve communications between developers and users, improve quality, and enhance knowledge transfer throughout the lifecycle of the SUT.

### ***3.3 Research Focus***

This dissertation identified software testing techniques and defined what capabilities from the software testing suite can be implemented in an MBSE construct to provide inputs to the flight test planning process and help AFOTEC realize the benefits of MBSE. Specifically, it contributes to model reusability, provides a trustworthy source for all stakeholders, and helps the flight test community support efficient acquisition planning. This research analyzed software testing methods and metrics to enable flight test planning of hardware and software testing using MBSE.

### ***3.4 Research Questions and Task List***

The following research questions address the proposed problem statement above.

#### ***3.4.1 Research Question 1***

##### **What software testing techniques are translatable to flight testing?**

The software and flight test communities have distinct requirements for conducting testing in their respective realms; this research question describes their similarities and differences to help better conduct testing or focus efforts to improve current testing approaches. To address this research question, this research effort attempted to identify translatable techniques that allow the flight test community to define and implement identified software regression testing techniques.

##### ***3.4.1.1 Research Question 1 Task 1***

**Perform a narrative review of the vocabulary used in software testing and map it to the vocabulary used in flight testing.**

This task focused on translating the concepts and definitions between the two different areas of knowledge and allowing testers from the flight test community to understand the notions from the software community and how they correlate. A review of software testing textbooks and a survey of scholarly articles was used to create a dictionary of commonly used software testing terms and their associated definitions. Using the AFOTEC's Analyst Test Guide and Measures Guide as a model, a glossary of commonly used flight-testing terms and their associated definitions was created to compare and correlate related terms within the two disciplines.

#### ***3.4.1.2 Research Question 1 Task 2***

**Identify the software testing techniques used in controlled software test and evaluation.**

This task reviewed the software T&E techniques applied to controlled testing. The task was to conduct a trade space analysis to realize the critical components, discover relationships, and make available methods to testers from the flight test community. The collection of software techniques was reviewed and analyzed to see if the flight test community could adopt the processes identified in flight testing. Examples of black box testing techniques that were extracted from the software testing literature include Boundary Value analysis, Orthogonal Array testing, Equivalence Partitioning, Fuzz Testing, Graph-Based Testing, and State Transition Testing, and examples extracted from grey box testing techniques include matrix and regression testing [18]. Only a few of these techniques have been applied to or considered in the context of flight testing, reinforcing the hypothesis that model-based flight testing could benefit from implementing software flight test techniques.

#### ***3.4.1.3 Research Question 1 Task 3***

**Examine which software testing techniques can be modified for implementation in flight testing.**

This task used the software techniques cataloged from the previous functions to examine which testing methodologies were directly applicable or modified for flight testing. This collection of software techniques was assessed quantitatively (using a Pugh Decision Matrix or similar) to see if any of the identified methodologies could be adapted to a valid flight-testing approach. The approaches were analyzed to see how much modification each technique required. These techniques were compared to a list of AFOTEC flight testing procedures to identify the intersection and union of testing techniques. The approach with the most negligible modification was selected for implementation in Research Question 2.

#### ***3.4.1.4 Research Question 1 Task 4***

**Identify the metrics used in software testing used in controlled software experimentation.**

Whereas the previous task focused on the vocabulary and concepts of software testing, this task created a collection of software testing metrics used by the selected software testing techniques from research question 1 task 2 as they apply to controlled testing. The software metrics were reviewed and analyzed to see if the flight test community can adapt these metrics to the current flight test planning process. Examples of metrics extracted from the software testing literature include test coverage, Defect Removal Efficiency, Requirement Volatility, Index, failed and executed test cases, and Requirement Traceability Matrix (RTM) [19]. These metrics were evaluated for their potential to support an improved flight-testing process, making testing more

accurate and efficient. These metrics were compared to the ones used in the AFOTEC flight testing procedures to identify the intersection and union of the test metrics. These metrics were then used to inform task 5.

#### ***3.4.1.5 Research Question 1 Task 5***

**Examine which metrics used in software testing are suitable for implementation in flight testing.**

This task involved a detailed examination of the metrics identified in research question 1 task 4. The goal was to determine which of these metrics were appropriate for use in flight testing. A survey conducted with a panel of subject matter experts was used to assess a SUT's operational effectiveness and suitability for combat operations under realistic operational conditions. This process ensured that critical operational issues were addressed and provided additional information on the system's operational capabilities. The new metrics identified were then used to answer research question 2.

#### ***3.4.2 Research Question 2***

**What are the costs and benefits of the techniques associated with research question 1 when implemented in an MBSE construct?**

##### ***3.4.2.1 Research Question 2 Task 1***

**Define an MBSE model in which the techniques from research question 1 can be realized.**

This task created a simplified MBSE model that was used as a case study. The model represented a training system to train new Air Force Operational Test and Evaluation Center (AFOTEC) members in conducting operational test and evaluation (OT&E). A set of MBSE



diagrams, such as mission domain elements, requirements, use cases, and activities, were created to support the test case. The model developed, demonstrated, and evaluated a process that generated the required information as inputs to the AFOTEC test planning processes.

#### ***3.4.2.2 Research Question 2 Task 2***

**Measure the degree of data reuse for the proposed MBSE plan integration.**

The literature on MBSE states that data reuse, or model reuse, is a crucial method through which MBSE achieves process improvements. Therefore, this task used the MBSE model developed in research question 2, task 1, and two other models to demonstrate a model reuse process that measured the reuse of information and model artifacts. The software techniques developed in research question 1 task 3 were applied to these models, and their performance was analyzed for their effect on model component reuse.

#### ***3.4.3 Research Question 3***

**Compare the techniques associated with research question 2 with the current approach used in flight testing.**

##### ***3.4.3.1 Research Question 3 Task 1***

**Characterize the current flight-testing approach and categorize the metrics implemented in flight testing.**

A narrative summary of the current AFOTEC approach clarifies how the flight test community understands and interprets the information collected and used to develop various flight test plans. The AFOTEC test planning process represents the baseline and state of the art against which the novel process was measured.

#### ***3.4.3.2 Research Question 3 Task 2***

**Compare the approach from research question 2 with the current approach defined in research question 3 task 1 and assess their benefits and costs in an example acquisition planning scenario.**

This task compared the approach from research question 2 with the current AFOTEC flight test approach to aid in and measure improvements in test planning. A survey was conducted using current AFOTEC personnel assigned to detachment 5 to assess the model-based test plan's usefulness, effectiveness, and suitability. Furthermore, the survey asked the participants to provide an improvement metric based on their assessment of the information provided by the model-centric test plan.

## CHAPTER 4 – LITERATURE REVIEW

### *4.1 Introduction*

Testing for software and flight testing focuses on systems or sub-systems to ensure that functions and capabilities have not been compromised or degraded and the stability and functionality of existing capabilities. Like acquisition systems, software constantly evolves by correcting previously unknown issues and incorporating new functionalities and capabilities. The software community performs testing when changes are made to the software to provide confidence that the recent changes do not change the behaviors of the current unchanged parts of the software [20]. Similarly, the acquisition community is constantly planning, executing, and reporting on system and subsystem testing to ensure that functions and capabilities have not been degraded and to confirm the stability and functionality of existing capabilities. Since software testing has the same goals as flight test, it behooves the flight test community to explore and possibly appropriate attributes from software testing and apply them for FOT&E regression testing.

Software testing focuses on identifying fault detection and safeguards against new defects being introduced into the extended code or specification changes. The ANSI/IEEE 1059 [1] standard defines software testing as analyzing a software item to detect the differences between existing and required conditions, such as defects, errors, or bugs, and to evaluate the features of the software item. Therefore, software testing usually consists of verification and validation. The following definitions are used in the software community to convey their intended meaning:

Verification is the process of determining whether the product fulfills the requirements. The question commonly asked is, “Are we building the product right?” [21].

Validation: This assessment ensures that products conform to their intended usage and meet customer expectations. The question commonly asked is, “Are we making the right product?” [21]. Verification is considered more technical since it relies on knowledge about the individual software artifacts, requirements, and specifications. Validation usually depends on familiarity with the application or operation for which the software is written. Other definitions used in the software community are that of a test requirement, which is a specific element of a software artifact that a test case must satisfy or cover, and coverage criterion, which is a rule or collection of rules that yield test requirements [22].

Software testing is a process that helps identify errors, faults, and failures in a system to make the software as efficient as possible. The purpose is to achieve a high level of reliability, maintainability, availability, security, survivability, portability, capability, efficiency, and integrity in the system [21]. Software testing is a way to guarantee that the system works as specified and provides as much functionality as specified. Software testing is of interest to developers, testers, organizations, and end-users [22].

#### ***4.2 Model-Driven Test Approach***

“Model-based testing (MBT) is the design of software tests using an abstract model that represents one or more aspects of the software” [22]. The model usually describes aspects of the intended behavior of the software. These models are often described with UML diagrams and sometimes produce anticipated outputs based on the behavior of the software created during the design stage of development. Tests are then derived from abstract structures or software behavior diagrams to establish input values that will effectively test software. Developing tests can be divided into four discrete tasks: test design, test automation, test execution, and test evaluation [22].

The model-driven approach to software development has garnered much attention from industry and academia. Model-based development using Unified Modeling Language (UML) diagrams, such as state machine diagrams, use-case diagrams, and sequence diagrams, has allowed many researchers and testers to generate test cases that are independent of any implementation of the design [23]. These ideas are the basis for a set of software development approaches that are known as model-driven development (MDD) that are used to define relationships between and define dependencies by which a solution was created and understand the implications of changes at any point in that process [24].

The Model-Driven Test Design (MDTD) process divides testing into tasks that simplify test generation to help test designers isolate tasks at a higher abstraction level and use mathematical engineering structures to design test values independently of the details of the software or design artifacts and test execution. The critical step to an MDTD test case design is to determine the primary factors in which a test successfully finds failures in the software. This process is shown in Figure 7, which shows test design activities above the line and other test activities below [22].

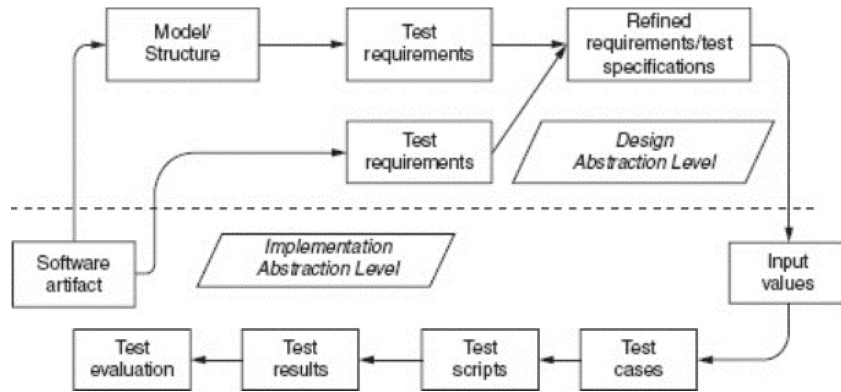


Figure 7. Model-Driven Test Design

MDTD lets test designers “raise the level of abstraction” so that a subset of testers can do the mathematical aspects of designing and developing tests. In contrast, traditional testers and programmers find values, automate, run, and evaluate the tests.

The starting point in Figure 7 is at the software artifact, which could be a program source, a UML diagram, or a user manual. Criteria-based test design uses the artifact to create an abstract model of the software in the form of an input domain, a graph, logic expressions, or a syntax description to create test requirements, while a human-based test design uses the artifact to consider problems in the software and then creates requirements to test for those problems [22]. The process creates test requirements that describe the input values that satisfy the requirements, bringing the process down from the design abstraction level to the implementation abstraction level.

Since the primary intent of Test-Driven Development (TDD) is to define the behavior of the software as opposed to evaluate whether the behavior is correct, which is the traditional role of testing, the major disadvantage of TDD is that it tends to focus on behavior that should happen under regular use and the tests developed are less likely to navigate confused-user paths or malicious-user paths, where users break through security barriers and abuse the software [22].

Test engineers oversee the technical test activities that include the design of test inputs, producing test case values, running test scripts, analyzing results, and reporting results, as illustrated in Figure 8.

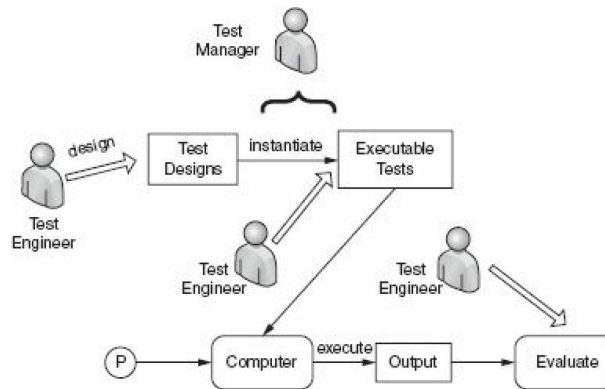


Figure 8. Activities of Test Engineers

Test engineers use requirements that are transformed into values and scripts that are used for execution. These executable tests are run against the software, denoted P in the Figure, and the results are evaluated to determine if the tests reveal a fault in the software [22].

Software engineers employ numerous techniques to address these challenges, the most foundational being using models and modeling. “Models provide abstractions of a physical system that allow engineers to reason about that system by ignoring extraneous details while focusing on the relevant ones. All forms of engineering rely on models as essential to understanding complex real-world systems” [24]. Since software testing is complicated and the goal of having the utterly correct software is unreachable, test engineers need to evaluate software based on its “behavior” to decide if it is acceptable within the consideration of many factors including reliability, maintainability, and efficiency [22]. Models predict system qualities, describe specific properties significantly when the system has changed, and communicate key system characteristics to various

stakeholders. The models can be created before implementing the physical system or derived from an existing or developing system to aid in understanding its behavior. [24].

Test cases can be designed with a “human-based” approach, where a test engineer uses domain knowledge of the software’s purpose and experience to create tests that will be effective at finding faults [22]. Human-based test design requires the tester to know the software’s application domain, testing, and user interfaces to create tests such that the software is stressed by providing very large or minimal values, boundary values, invalid values, or other values that the software may not expect during typical behavior, as well as explicitly consider actions the users might do, including unusual actions.

Test cases can also be designed to satisfy well-defined engineering goals such as coverage criteria. Formal coverage criteria, a tool to decide what test inputs to use during testing, facilitate the likelihood of detecting problems in a program and guarantee that the software is of high quality and reliability by creating abstract models and manipulating them to design high-quality tests. These tests use requirements, specifications, design artifacts, or source code to develop test cases for execution. Coverage criteria provide a structured, practical way to search the input space with little overlap. Thus, the main advantage of using coverage criteria is the improvement of quality and reduction of the cost of test data generation by maximizing the “bang for the buck” with fewer tests that are effective at finding more faults [22]. More importantly, coverage criteria can provide stopping rules to know when to stop testing.

### ***4.3 Software Testing Methodologies***

After the code is developed, it is mandatory to test the software to identify and debug all errors before it is released. However, it is impossible to identify and debug all the errors in the



software at every phase. Although testing helps to find mistakes and bugs, it cannot conclude that the software is bug-free [21]. Figure 9 illustrates a hierarchy of software testing methodologies.

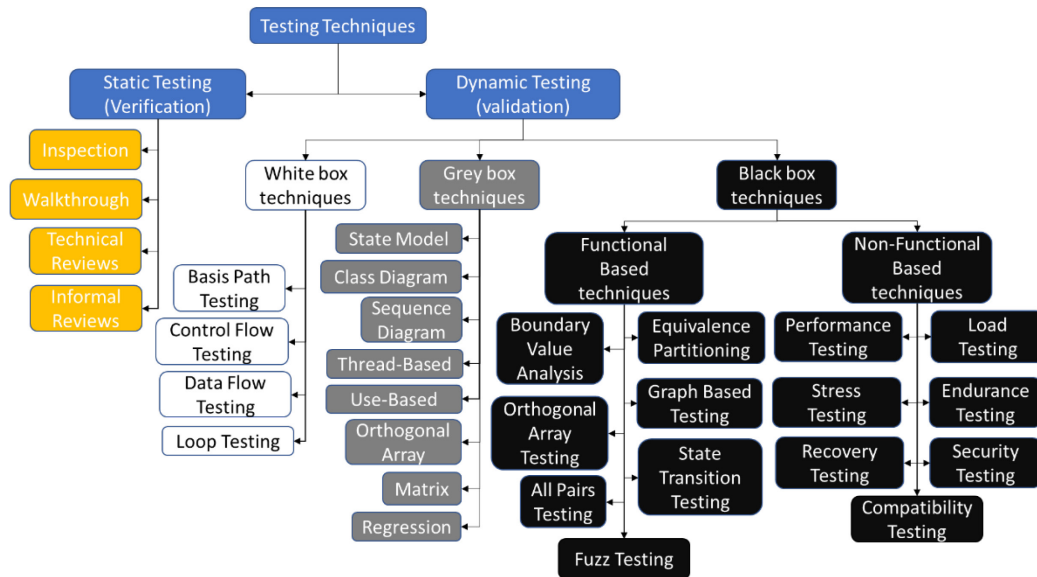


Figure 9. Hierarchy of Software Testing Methodologies

Software testing techniques can be categorized into two major groups: static and dynamic testing. The types of procedures employed in each group can then define each group.

#### 4.3.1 Static Analysis and Dynamic Testing

Software testing techniques are generally categorized into static and dynamic testing. Static testing, also known as manual testing, refers to the method of testing where no code is executed. It is part of verification testing and usually starts with the initial phase of the Software Development Life Cycle (SDLC), where the main objective is to enhance the quality of the software product by helping identify and resolve errors early in the software development process [21]. Static testing reviews documents such as Software Requirement Specification (SRS), design documents, source code, test suites, and web page content and is accomplished before code

deployment. Static testing provides for the evaluation of code as well as documentation and includes techniques such as [21]:

Inspection: This is primarily done to locate defects. Moderators conduct the code walkthrough. This type of formal review prepares a checklist to check the working document.

Walkthrough: This is not a formal process. The authors lead it. The author advises the participants through the document according to their thought processes to achieve a common perception. It is helpful for higher-level documents like requirement specifications, etc.

Technical Reviews: A review is conducted to ensure that the code aligns with technical specifications and standards, which may encompass test plans, test strategies, and test scripts.

Informal Reviews: This is the static testing technique in which the document is reviewed unofficially, and helpful comments are implemented.

Dynamic testing, also known as automated testing, is a software testing technique in which the code's dynamic behavior is analyzed. It falls in the realm of validation testing, where the actual system is considered. Dynamic testing encompasses methods for assessing or approximating software quality by executing it with actual data and under real or simulated conditions. [25]. Dynamic testing encompasses testing the software with various input values and observing and analyzing the output values. This type of testing is divided into two groups: white box and black box testing. A third category, referred to as grey box testing, combines both box and black box testing.

#### ***4.3.2 White Box Testing***

White box testing, also known as structural, code-based, precise, or glass box testing, is a testing process that focuses on examining the internal logic and structure of source code. This

testing process uses the internal logical arrangement of the software and can test all the independent paths of a module; every logical decision is exercised, all loops are checked at each boundary level, and internal data structures are exercised [18]. To properly design and test the executable source code, a tester must know the source code and the application's internal workings. The most widely acknowledged white box software testing techniques are as follows:

Control Flow Testing: A structural testing strategy uses the program control flow as a model control flow and favors more straightforward paths over fewer but complicated ones [26].

Basis Path Testing: This type of testing allows the test case designer to produce a logical complexity measure of procedural design and then uses this measure to outline a basic set of execution paths [26]. Each independent path of code is selected for testing and is represented with flow graphs to characterize the control flow of the program [18].

Data Flow testing: In this type of testing, the control flow graph is annotated with information about how the program variables are defined and used [26].

Loop Testing: This technique focuses on the validity of the loop construct since most errors occur at the beginning or end of a loop and search for the possibility that the loop successfully ends [18].

Some of the advantages of white box testing include [27]:

- All independent paths in a module are exercised at least once.
- All logical decisions are exercised.
- All loops at their boundaries are executed.
- Internal data structures are exercised to maintain their validity.
- Errors in hidden codes are revealed.

- Approximate partitioning is done by execution equivalence.
- The developer carefully gives reasons for implementation.
- Maximum coverage is attained during test scenario writing.

The disadvantages of white box testing are [27]:

- Skilled testers are needed to carry out testing because knowledge of the internal structure of the code is required.
- Many paths will remain untested as it is tough to investigate every bit of code to find hidden errors, which may create problems, failing the application.

#### ***4.3.3 Black Box Testing***

Black box testing is conducted when testers do not have knowledge of internal design or access to the source code of the application or system, have some knowledge of the system architecture, and have access to requirements specifications or requirements. The testers only understand what the software is supposed to do but not how it does it. Thus, testers only know the set of inputs and specific outputs that describe the application, and this set of techniques is used to ensure that the application or system recognizes all inputs needed by the system and provides the correct output [18]. The most common black box testing methods used are:

Equivalence Partitioning: This technique divides the input domain of a program into equivalent classes from which test cases can be derived. Equivalence partitioning can be used to minimize the number of test cases [21].

Boundary Value Analysis: It targets the testing at borders or where the extreme boundary values are chosen. It comprises minimum, maximum, just inside/outside boundaries, error values, and typical values [21] [26].

Fuzz Testing: This approach provides random input to the application using malformed or semi-malformed data injection in an automated or semi-automated session and is used for characterizing implementation bugs [26].

Orthogonal Array Testing: Orthogonal array testing is a statistical way of sampling the input domain to reduce the number of test combinations. Columns and test cases represent rows that represent the variables. It is used when the input domain is minimal, but too extensive of an input domain can lead to exhaustive testing [26].

Cause-Effect Graph: This testing technique begins by generating a graph and establishing the relation between effect and its causes [26].

All Pair Testing: This technique is designed to execute all possible combinations of each couple of input parameters where the objective is to perform all possible combinations of each couple of input parameters [18] [26].

State Transition Testing: This testing approach is used in testing state machines and navigation of graphical user interfaces and is used to capture several types of system requirements and documented internal system design [18] [26].

Some of the advantages of black box testing include [18] [26]:

- Testers do not need to know specific programming languages.
- Testing is done from a user's point of view.
- It helps to discover any ambiguities or inconsistencies in the requirement specifications.
- Programmers and validators are independent of each other.
- By achieving reasonable testing, the number of test cases can be reduced.

- Well-suited technique and most efficient for large code segments.
- No code access is required.

The disadvantages of black box testing are [18] [26] [27]:

- Test cases are complex to design without precise specifications or requirements.
- Chances of having a repetition of tests that the programmer already does.
- Some parts of the back end have not been tested at all.
- There is some chance of having unidentified paths during this testing; only small numbers of possible inputs can be tested.

#### ***4.3.4 Grey Box Testing***

Grey box testing is a software technique combining the benefits of black and white box testing. The tester has limited knowledge of the internal structure, such as internal logic or underlying code, and an understanding of fundamental aspects of the system [26]. It sometimes requires reverse engineering of the application or system to work out the boundary values [21]. Verification of both the output of the user interface and the process that leads to that user interface output is tested [26]. The gray box testing methodology is used to test software applications. It heavily depends on a host platform debugger to execute and validate the software under test [27]. The most widely known grey box software testing techniques are as follows:

State-Model-Testing: It examines each method of an object, transition, and transition paths at each stage of an object [21].

Class-Diagram Testing: It examines all the derived classes of the base class [21].

Sequence-Diagram Testing: It examines all the methods occurring in the sequence diagram [21].

Thread-Based Testing: This approach takes all the classes of single-use cases and integrates them into a test. This approach continues until all the paths of all use cases have been considered [21].

Use-Based Testing: This testing divides the classes into services that need something from other courses or into services that do not need something from other courses [21].

Orthogonal Array Testing: This testing technique is a systematic and statistical method that uses a subset of all possible combinations [26].

Matrix Testing: In matrix testing, the status report of the project is stated [26].

Regression Testing: If new changes are made in software, regression testing implies running test cases [26].

Pattern Testing: Pattern testing verifies the excellent application for its architecture and design [26].

Some of the advantages of grey box testing include [18]:

- This testing technique is implemented from the user's point of view rather than from the designer's point of view.
- The tester designs efficient test scenarios.
- Whenever possible, both the benefits of white box and black box testing should be given.
- Developers carefully give reasons for implementation.

Some of the disadvantages of grey box testing include [26]:

- Test coverage is limited because of the unavailability to access source code.

- Many program paths remain untested.
- The test cases can be redundant.

#### ***4.3.5 Nonfunctional Testing Techniques***

Nonfunctional testing is accomplished to test various attributes of a system, such as stress and load, and is concerned with the requirements not related to the application's functions but rather to assess the readiness of a system according to the various criteria not covered by functional testing [21]. The following techniques are employed in conducting nonfunctional testing [21].

Performance Testing assesses and evaluates the entire system's performance under a specific workload.

Load testing: A load test is performed to ensure the load takes the capacity of a particular order. Load testing determines the system's behavior under normal and peak load conditions.

Endurance testing determines the system's behavior after a particular time. For example, a method may work perfectly fine in the initial first hour, but its performance decreases after three hours of execution.

Stress Testing: It is performed to determine the system performance beyond standard operational capacity, often to a breaking point. It is concerned with the system's load-taking ability.

Security Testing: Security testing assesses whether the system is safe. It is a process that concerns whether an information system protects data and maintains functionality as intended.

Recovery Testing: Recovery testing checks the system's recovery after a crash or hardware failure. In this type, the software is forced to fail under a given circumstance, and the recovery is subsequently tested.



Compatibility Testing: Compatibility testing is concerned with checking the system's compatibility with the rest of the environment. It checks the harmony of the developed system with the various other components such as hardware, additional software, database management system (DBMS), and operating system.

#### ***4.3.6 Summary***

The knowledge gained during this literature review is that Model-based testing (MBT) uses abstract models to represent aspects of the system and describe the intended behavior by using UML diagrams; we can test abstract structures or behavior diagrams using a Model-Driven Test Design (MDTD) process to determine the factors for a successful test. Furthermore, software testing, which uses static and dynamic testing to perform verification and validation of software to achieve a high level of reliability, maintainability, survivability, and integrity in the system, is categorized into white, black, and grey box testing, which are dependent on the knowledge and skill of the tester. Ultimately, testing may also evaluate different aspects of a system, such as stress, which are unrelated to the application's functions. This evaluation, known as non-functional testing, assesses the system's readiness. Since the goal of both software and flight testing is to ensure that the functions and capabilities of a SUT have not degraded and safeguard the stability and functionality of existing capabilities, this comprehensive view has provided the ingredients needed to apply concepts from software and model-based testing as inputs into the next stage of this dissertation.

## CHAPTER 5 – DEVELOPING MODEL-BASED FLIGHT TEST SCENARIOS

### ***5.1 Introduction***

This chapter addresses Research Question 1, which is restated as follows: *What software testing techniques are translatable to flight testing?*

To answer this research question, the chapter begins with section 5.2, which presents a narrative review of the vocabulary used in software testing and maps the language used in flight testing. Section 5.3 then identifies the types of software testing techniques used in controlled software experimentation and examines which software testing techniques can be modified for implementation in flight testing in the context of a CSARL grey box testing case study. Section 5.4 identifies the metrics that can be adopted from software testing to controlled software experimentation and assesses the suitability of those metrics for implementation in flight testing.

### ***5.2 Preliminary Work***

The following discussion contains the effort completed before my preliminary examination. This information was presented to my dissertation committee during the initial assessment, which detailed the work accomplished to answer research question 1, task 1. The discussion focuses on the types of software techniques used in the software testing domain and provides the background research required to prepare for my first publication. The results of this research illustrated the commonalities between flight testing and software testing. Furthermore, the results compared the terminology, techniques, and material that can be ported over from software testing to flight testing.

### 5.2.1 Research Question 1 Task 1

While performing a narrative review of software testing textbooks and scholarly articles, a vocabulary of terms used in software testing was collected. This provided knowledge of terminology and helped define the concepts in software testing. The findings were captured in a document containing the collected and its associated definition. Similarly, terms commonly used in flight tests were collected, and the same treatment was applied. Related terms between the two disciplines were sorted, correlated, and arranged in an N-by-N matrix. Table 3 The table illustrates a sample of the basic terminology used in both disciplines and where it falls in the MBSE construct. It demonstrates the similarity between flight and software testing disciplines.

Table 3. N-by-N Matrix of Software Testing Vocabulary and Flight-Testing Vocabulary

The matrix classifies the vocabulary from software and flight testing to show their typical relationship. It allows for a common understanding between the two disciplines and how they relate to the MBSE construct.

| Function                        | Software Traditional | Flight Test Heritage |             | Model-Based Systems Engineering |
|---------------------------------|----------------------|----------------------|-------------|---------------------------------|
|                                 |                      | Developmental        | Operational |                                 |
| Define test Requirements        | Test Criterion       | Specific Objective   | MOP/MOS     | Operational Viewpoint           |
| Test to fulfill the requirement | Verification         | General Objective    | COI         |                                 |
| Design the Experiment           | Condorcet Techniques | DOE                  | DOE         | Logical/Functional Viewpoint    |
| Test planning                   | Test Case            |                      |             | Physical Viewpoint              |
| Feedback from Test Data         | Variable             | Variable             | Variable    |                                 |

Furthermore, using the hierarchy structure described in Figure 9, Table 4 was constructed as an example to categorize the types of interaction between flight and software testing techniques, highlighting which software techniques could be used to help answer research question 1, task 3.

Table 4. N-by-N Matrix of Software Techniques

The matrix classifies the software testing techniques available and is categorized by the type of box testing the methods fall into. The check mark indicates which technique can be applied to the testing scenario. The asterisk indicates that these techniques are performed as laboratory tests and are not usually associated with flight testing. At the same time, the X denotes that the method does not apply to flight testing.

| Type of Testing | Function     | Software Testing Technique |                      |                          | Has been Applied to Flight Test? |    |
|-----------------|--------------|----------------------------|----------------------|--------------------------|----------------------------------|----|
| Static          | Verification | Inspection                 |                      |                          | ✗                                |    |
|                 |              | Walkthrough                |                      |                          | ✗                                |    |
|                 |              | Technical Review           |                      |                          | ✓                                |    |
|                 |              | Informal Review            |                      |                          | ✓                                |    |
| Dynamic         | Validation   | White Box                  | Basis Path           |                          | ✗                                |    |
|                 |              |                            | Control/Data Flow    |                          | ✗                                |    |
|                 |              |                            | Loop                 |                          | ✗                                |    |
|                 |              | Black Box                  | Functional Based     | Boundary Value Analysis  |                                  | ✓  |
|                 |              |                            |                      | Orthogonal Array         |                                  | ✓  |
|                 |              |                            |                      | All Pairs                |                                  | ✓  |
|                 |              |                            |                      | Fuzz                     |                                  | ✗  |
|                 |              |                            |                      | Equivalence Partitioning |                                  | ✓  |
|                 |              |                            |                      | Graph-Based              |                                  | ✗  |
|                 |              |                            |                      | State Transition         |                                  | ✓  |
|                 |              |                            | Non-Functional Based | Performance              |                                  | ✓  |
|                 |              |                            |                      | Stress                   |                                  | ✓  |
|                 |              |                            |                      | Recovery                 |                                  | ✓* |
|                 |              |                            |                      | Compatibility            |                                  | ✓  |
|                 |              |                            |                      | Load                     |                                  | ✓* |
|                 |              |                            |                      | Endurance                |                                  | ✓  |
|                 |              |                            |                      | Security                 |                                  | ✓* |
|                 |              | Grey Box                   | State-Model          |                          | ✓                                |    |
|                 |              |                            | Class-Diagram        |                          | ✓                                |    |
|                 |              |                            | Sequence-Diagram     |                          | ✓                                |    |
|                 |              |                            | Thread-Based         |                          | ✓                                |    |
|                 |              |                            | Use-Based            |                          | ✓                                |    |
|                 |              |                            | Orthogonal Array     |                          | ✓                                |    |
|                 |              |                            | Matrix               |                          | ✗                                |    |
|                 |              | Regression                 |                      | ✓                        |                                  |    |

As illustrated in Table 4, the green check marks indicate that the software method is already being used, and existing documentation and processes facilitate its implementation in flight testing practice. The yellow check marks signify that the related software procedure is also being used by the flight test field, but not to the extent in which software testing uses them due to limiting factors

such as the number of samples required to accomplish the task or the lack of a proper environment in which to test the SUT due to either constraint on assets needed for testing or the type of testing that might be sensitive in nature. The red check marks denote the practices that are not common or implemented by the flight test community and thus were not analyzed and evaluated for application to flight testing.

As a result of this work, new knowledge was developed to define a comprehensive set of software and flight-testing methods and what similarities exist between these two separate testing applications. For example, one of the results of this process was the discovery of an opportunity to leverage many model-based testing procedures and concepts developed in software testing to the flight-testing regime. These procedures include state-model testing, class-diagram testing, thread-based, and use-based testing, all of which fall under the category of MDTD. These findings support the concept that MBSE-enabled T&E would enable innovations in model-based testing applied to flight systems.

This process has revealed the available methods and identified techniques adaptable for transplanting into flight test practices (as indicated by the red check marks). It aimed to develop novel approaches for testing SUTs within the discipline of flight testing. These initial findings laid the groundwork for evaluating the software testing techniques most suitable for integration into flight test procedures.

### ***5.3 Publication Details***

The following discussion contains the work from Reference [28] which was published in the International Test and Evaluation Association (ITEA) Journal of Test and Evaluation and presented at the Defense and Aerospace Test and Analysis Workshop (DATAWorks) conference in Alexandria, Virginia. DATAWorks is sponsored by a multi-organization collaboration between

the Director of Operational Test & Evaluation (DOT&E) within the Office of the Secretary of Defense, National Aeronautics and Space Administration (NASA), the Institute for Defense Analyses (IDA), and the Section on Statistics in Defense and National Security (SDNS) of the American Statistical Association (ASA). The work is reproduced within this dissertation but has been reformatted to meet the dissertation style guidelines. The references within the publication have been incorporated into the reference section of this dissertation, but the content has not been altered. The publication details the work accomplished to answer research question 1, task 2, and 3, along with research question 2, task 1.

### ***5.3.1 Abstract***

The Department of Defense (DoD) is undergoing a digital engineering transformation in every process of the systems engineering lifecycle. This transformation requires implementing DoD Test and Evaluation (T&E) processes and executing model-based testing methodologies. This paper describes and assesses a grey box model-driven test design (MDTD) approach to create flight test scenarios based on model-based systems engineering artifacts. To illustrate the methodology and evaluate the expected outcomes of the process in practice, a case study using a model representation of a training system utilized to train new Air Force Operational Test and Evaluation Center (AFOTEC) members in conducting operational test and evaluation (OT&E) is presented. The results of the grey box MDTD process are a set of activity diagrams that are validated to generate the same test scenario cases as the traditional document-centric approach. This paper will use artifacts represented in System Modeling Language (SysML) to discuss key comparisons between the conventional and MDTD processes. This paper demonstrates the costs and benefits of model-based testing and their relevance in the context of operational flight testing.

### ***5.3.2 Introduction***

The primary objective of DoD T&E is to provide decision-makers with essential information for assessing the suitability of a weapon system for fielding [13]. The T&E community must deliver defensible results that inform senior leaders about system performance and the impact of design parameters on effectiveness and maintainability. Traditional T&E flight test planning, conducted by the U.S. Air Force, follows a systematic, sequential document-centric, and human expert-driven approach integrated throughout a system's acquisition life cycle [29] [30] [31]. Flight test planning ensures the accuracy and reliability of data gathering that meets requirements for verification, validation, and thoroughness of system integration but relies heavily on document-centric procedures, system matter experts (SME) process knowledge and technical expertise, and a sequential flow of approvals. Comprehensive safety and test plans detail the technical aspects and logistics necessary to execute and report flight, ground, and laboratory test results for air vehicles, subsystems, and components [13].

While well-established, the current T&E approach suffers from three significant drawbacks: 1) the need for constant manual updates to documents due to ongoing changes throughout the development lifecycle, 2) unwieldy and document-centric communication among stakeholders, and 3) the lack of ability to insert new technology or operational requirements due to the manual updates of documents and the requirement for multiple meetings of SMEs [4]. The process of devising test and safety plans is isolated from the design, development, and operation of the system under test (SUT). This creates interdependent serial processes that result in long timelines for submission and approval, which are resistant to requirement changes. Developing a test plan can involve multiple numbers of SMEs and years of effort. Test products are developed, discussed, and refined through multiple meetings and iterations between SMEs, experienced

testers, and operators, ensuring all system behaviors are identified and addressed [32]. Experienced test engineers must possess intimate knowledge of the system and develop test scenarios that accurately identify the relevant measures of performance/suitability (MOP/S) for achieving mission objectives and conveying information about the system's structure. However, multiple meetings requiring these experienced engineers drive significant resources and extended timelines. These experienced test engineers require significant resources, extending timelines and creating isolationist thinking that is resistant to changes in requirements. The outcome of these activities is a set of detailed test plans and safety packages, presented in document form, which outlines the resources and test activities planned for the SUT to generate information for decision-makers.

### ***5.3.3 Motivation***

Dr. Will Roper, the Air Force for Acquisition, Technology, and Logistics Assistant Secretary, emphasized that embracing digital design technologies would lead to a faster, more agile, and more competitive weapons procurement process, departing from the traditional defense acquisition approach [14]. In acknowledgment and response, the DoD released the Defense Digital Engineering Strategy in June 2018 to modernize the entire lifecycle of system design, development, delivery, operation, test, and sustainment within the Department [15]. The flight test community is now charged with integrating model-based software engineering (MBSE) concepts and processes seamlessly as part of a community-wide paradigm shift toward model-based test engineering methodologies [4]. By aligning with MBSE principles, the flight test community seeks to enhance the efficiency and effectiveness of flight testing, gain a deeper understanding of system behavior, and foster better collaboration among stakeholders [14] through model-driven test planning. This transition has the potential to do four things: 1) aid in the management of program complexities by highlighting communication routes and easing data transfer, 2) help in the



integration of new technologies and design approaches by enabling data-based decision-making, 3) enable re-planning on the fly to support changing mission scenarios, and 4) support the needs of stakeholders by tracking requirement flow-down [30] [33] [34].

In response to this declared need, this paper presents the creation of an MBSE framework that has been applied to the physical testing of a SUT that creates and defines test scenarios. A simple case study exemplifies the methodology and demonstrates the effectiveness of model-based flight test scenario development.

### ***5.3.4 Background***

#### ***5.3.4.1 Model-Based Testing***

The MBSE methodology encompasses a range of interconnected processes, methods, and tools that operate within a "model-based" or "model-driven" framework [35]. Various industries, including aerospace and software development, have embraced MBSE methodologies to streamline the creation of complex systems and products, including testing [36]. MBSE methodologies create models that are abstract representations of reality [37]. These artifacts can represent various aspects of the desired behavior of a physical SUT. Interconnected state machine diagrams, use-case diagrams, sequence diagrams, and mathematical models allow researchers and testers to generate independent test cases that are decoupled from specific design implementations [38] [23]. The model can be used to define test parameters further and generate the details of a test plan for a given test case, allowing for testing, investigation, and optimization without using the physical asset.

#### ***5.3.4.2 Model-Based Testing Techniques***

Models serve as essential abstractions of physical systems, enabling engineers to reason effectively by focusing on relevant details while ignoring extraneous complexities. All branches of engineering heavily rely on models to comprehend intricate real-world systems [24]. Model-based testing allows test engineers to concentrate on evaluating software behavior models to determine their acceptability, considering factors like reliability, maintainability, and efficiency [22]. These tests may be developed as a preliminary step before implementing the physical system, or they may stem from an existing or in-development system [24].

This study adopts the philosophies of grey box testing to realize maximum value and compatibility with current operational flight test procedures. In grey box testing, the tester combines the benefits of black and white box testing. White box testing involves a meticulous examination of the internal logic and structure of the source code, necessitating the tester to possess comprehensive knowledge of the source code and the application's internal workings [18]. Black box testing is conducted when access to internal design or source code is not possible. Black box testers only understand the intended functionality of the software and have access solely to requirements specifications [18]. Grey box testing combines the benefits of both black box and white box testing, where the tester possesses limited knowledge of the internal structure or underlying code and understands the fundamental aspects of the system [26]. Furthermore, non-functional testing can be performed to assess various attributes of a system, such as stress testing, which is not directly related to the application's functions. Non-functional testing can evaluate the system's readiness [26].

All these testing techniques require an underlying process for executing the methods. Among the most common methods used for test case generation from SysML diagrams is coverage

criteria. Coverage criteria are defined as the set of conditions and rules enforcing a set of test requirements on a software test to enhance the generation of comprehensive test cases based on the number of elements to cover or visit within a diagram [39]. There are eight standard transition-based coverage criteria used in test case generation, of which all-paths coverage specifies that each executable path should be followed at least once, and all-configuration coverage is required to visit every configuration of the MBSE models at least once [39].

Leveraging some of these model-based testing techniques and tools to plan flight tests, the T&E community can gain new approaches to evaluate flight test programs better, verify system functionality, ensure compliance with specifications, and ascertain that SUTs can consistently deliver the intended performance in operational scenarios.

#### ***5.3.5 The Flight Test Application***

Traditional flight testing commences once a program manufactures a representative prototype SUT. During this stage, developmental T&E (DT&E) tests the SUT for integration, interoperability, supportability, safety, and utility [11]. DT&E demonstrates that the engineering design and development process is complete, design risks have been minimized, and system specifications have been met [9]. DT&E flight testing is primarily overseen by the Air Force Flight Test Center (AFTC) which assembles and authors comprehensive test plans "to communicate technical detail and logistics required to execute and report results of flight, ground, and laboratory tests of air vehicles, subsystems, and components" [16]. AFOTEC is responsible for conducting integrated operational T&E (OT&E) throughout program development, with dedicated OT&E after the program enters Low-Rate Initial Production [29] [31] [11].

### **5.3.6 Methods**

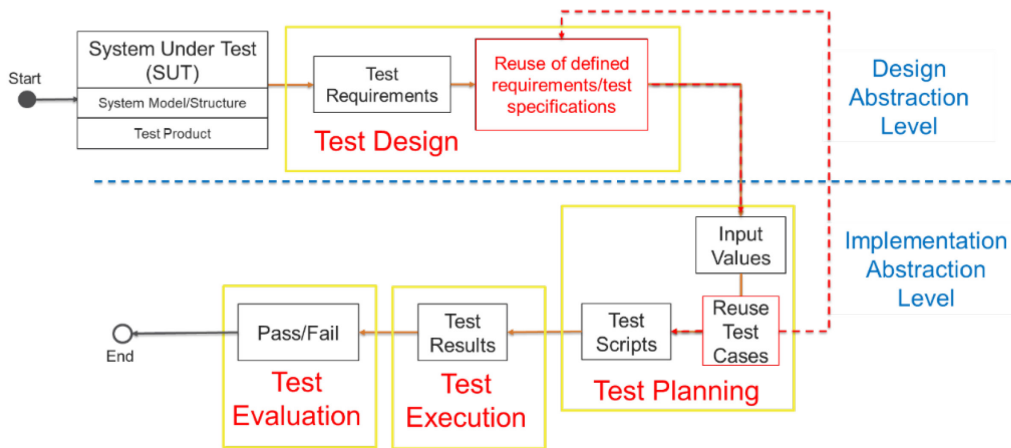
#### **5.3.6.1 Model-Driven Test Design**

By leveraging approaches such as MDTD and the capability to isolate and test individual components independently, a powerful approach to optimizing system test design can be realized. Through MDTD, relationships and dependencies that govern the creation of solutions ensure a comprehensive understanding of the repercussions of any changes made at any stage of the process by highlighting the broken relationships [24]. MDTD has demonstrated the ability to aid in efficient testing and significantly enhance the overall development process of software [24]. By embracing MDTD principles, flight test organizations may be able to achieve higher levels of precision and confidence in their testing and development endeavors [26]. Applying MDTD approaches to flight test has the potential to identify a system's reliability, maintainability, and survivability deficiencies and allow system designers to correct deficits quickly, thereby enhancing overall system quality [24].

The MDTD process empowers test designers to define tests at higher levels of abstraction by simplifying test generation into a series of tasks. The tasks use mathematical engineering structures to define test values independent of the intricacies of the design artifacts present in the software. The abstraction of the MDTD process allows the early involvement of designers in the test planning process and incorporates the testing into the design [22]. The canonical MDTD process [22] (illustrated in Figure 10 by the smaller black boxes) initiates with a test product, which could be a design element, a SysML model, a user manual, or actual source code within a SUT model [22]. The design artifact can take the form of a graph, logic expression, or a syntax description establishing test requirements [22]. MDTD empowers test designers to "raise the level of abstraction," [22] allowing a subset of testers to handle the high-level, statistical, and

defensibility aspects of test design and development. Traditional testers and programmers can then focus on automating tests, collecting and measuring values while executing the tests, and evaluating the results. This division of tasks enhances efficiency and effectiveness in the testing process by accomplishing both design and test planning in parallel and enhancing communications between designers and test stakeholders [22].

The flight test planning process can take advantage of the parallelism afforded by the MDTD process directly, as depicted by the yellow boxes in Figure 10. In flight testing, the design artifact can be defined as a requirement, function, or capability to be evaluated. Flight test design evaluates a design artifact and develops test scenarios at the design abstraction level. The test scenarios create detailed test plans at the implementation abstraction level that will be executed to collect the data required to evaluate the design artifact. The superimposed process results in test requirements that test the design artifact through the detailed test plan, bridging the gap between the design and implementation abstraction levels.



### 5.3.7 Existing Test Scenario Generation Process

#### ***5.3.7.1 Innovative Test Scenario Generation Process***

The importance of test scenarios cannot be overstated, as they play a pivotal role in identifying and communicating the conditions to be implemented during testing. They serve as a critical means to verify the successful and acceptable implementation of product requirements [40]. Test scenarios are devised to determine the essential measures (MOP/S) required to verify the successful and acceptable implementation of product requirements [40]. The MOP/S aligns with the mission objectives for the warfighter and provides insights into the system's structure and usability in the field [32]. Using an MBSE structure through various diagrams, the generation of test scenarios can be realized, facilitating the initiation of the testing process parallelizing with design and guiding the testing methodology [16].

A practical approach to creating test scenarios can be realized by beginning with the use case diagram to generate scenarios, followed by the activity diagrams to define flows, and finally, the sequence diagrams to provide the detail required by the test plan. By implementing grey box testing techniques borrowed from software implemented within an MBSE construct, the result provides an alternative to the document-centric test planning process. The process is illustrated in Figure 11. As a result, these diagrams guide developers in designing the system and provide testers with essential information for conducting thorough testing [16].

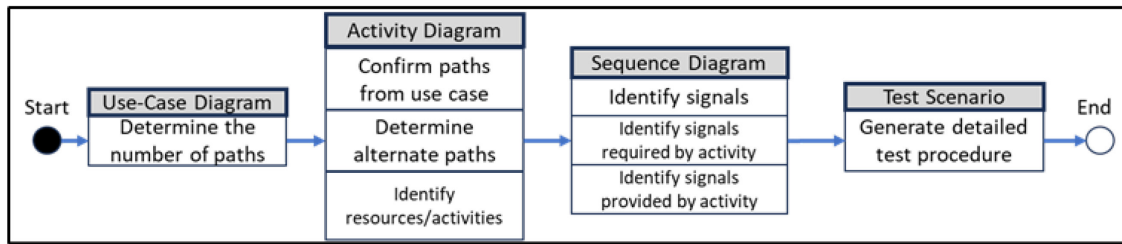


Figure 11. Grey Box Implementation of the Test Scenario Generation Process

Borrowing from the MDTD process and using the grey box test scenario generation process illustrated in Figure 11, along with the use case diagram depicted in Figure 13, a basic flow can be generated using the use case diagram to model a series of dialogs. The generation of test scenarios relies significantly on the flow of events, which can be divided into two parts: the main flow of events and the alternate flows of events that can be extracted from either the use-case or activity diagram. The main flow of events represents the "normal" happenings when the task is performed, while the alternate flows of events encompass optional or exceptional case flows relative to normal behavior [20]. Some alternate flows may return to the basic flow of events, while others designated as exceptions may reach an endpoint. Both the basic flow of events and the alternate flows can be further organized into steps or sub-flows [40]. A SysML diagram requires significant detail to fully specify a use case and define the flows between the system and its actors. These details can be developed into steps, explaining the actions of both the actor and the system in response that are realized in the number of unique test scenarios [40].

Test scenarios are then designed using a structured and practical approach that explores the input space with minimal overlap. Reduction in overlap aids in cost reduction by maximizing the effectiveness of tests [22]. Using the coverage criteria approach and a proper SysML model with sufficiently detailed use-case, activity, and sequence diagrams, we can apply coverage criteria techniques to derive test scenarios.



### ***5.3.8 Case Study Using Grey Box Test Scenario Generation Process***

To execute and assess the proposed MDTD processes, the following case study was produced using SysML implemented in MBSE with the Dassault Systemes' Cameo Systems Modeler™ software package following the method illustrated in Figure 11. The case study model was created from a design and development point of view that did not consider any inputs from the testing community. The model was developed by one person over approximately four months using the system description document (SDD), capability design document (CDD), and initial capabilities document (ICD) products, while the document-centric test plan and its associated process required multiple SMEs and a timeline of approximately six months to accomplish.

This case study focuses on AFOTEC's Combat Search and Rescue Locator (CSARL) system. CSARL is a fictional system developed as a training case for flight test personnel learning to create test plans. It is well documented and has a thorough reference test plan that has been redeveloped. This study will be an example system to demonstrate the proposed flight-test relevant test plan generation methods. Since the system is well documented and all AFOTEC personnel are familiar with the material, it makes a good benchmark for applying the grey box-inspired testing technique implemented in an MBSE environment to create a flight test scenario procedure without direct SME intervention. Moreover, it provides a means to compare outputs between the model-centric and document-centric processes.

The SUT for the case study is the CSARL system, which is a portable hand-held Global Positioning System (GPS) receiver system offering enhanced digital moving maps and real-time navigational capabilities for downed aircrew and search and rescue (SAR) forces during combat and peacetime survival scenarios. Its purpose is to complement current GPS-integrated survival radios, enhancing existing survival and rescue navigation capabilities without relying on

traditional map-and-compass techniques [41]. Documentation used in planning and testing the system, such as the CDD, ICD, SDD, and detailed test plan (DTP), illustrate the existing test scenario generation process. Figure 12 is an excerpt from the document-centric DTP and shows the table of contents highlighting the major headings of the information contained in the document.

| CSARL                                                        |    |
|--------------------------------------------------------------|----|
| Detailed Test Procedure                                      |    |
| <b>Contents</b>                                              |    |
| Scenario 1: Rescue Team Navigation and Recovery .....        | 3  |
| Build RCC & IP Waypoints .....                               | 4  |
| Plot Rescue Coordination Center (RCC) .....                  | 4  |
| Plot Isolated Personnel (IP) Waypoint .....                  | 4  |
| Plot Dry Run route from RCC to IP .....                      | 5  |
| Plot Drop-Off Point (DOP) .....                              | 5  |
| Build RCC to DOP Route.....                                  | 6  |
| Create Route from RCC to DOP. ....                           | 6  |
| Create Route from DOP to IP.....                             | 6  |
| Download Route from MPC to HHR.....                          | 6  |
| Select 'RCC to IP Route .....                                | 8  |
| Select Drop Off Point (DOP) Route.....                       | 8  |
| Navigate to IP .....                                         | 9  |
| Scenario 2: Isolated Personnel Navigation and Recovery ..... | 10 |
| Navigate from IP to PUP.....                                 | 11 |
| Scenario 3: Zeroize HHR.....                                 | 12 |
| To erase waypoints .....                                     | 12 |
| To erase routes.....                                         | 12 |
| Power down HHR.....                                          | 12 |

Figure 12. Excerpt of Table of Contents from CSARL Detailed Test Plan

To implement the MBSE approach, a SysML model was created using the Dassault Systemes' Cameo Systems Modeler™ software to represent the existing CSARL system and incorporate the system requirements from the SDD and descriptions defined in the ICD and the CDD. Figure 13 depicts the system's use case diagram and Figure 14 and Figure 15 illustrate the activity and sequence diagrams, respectively.

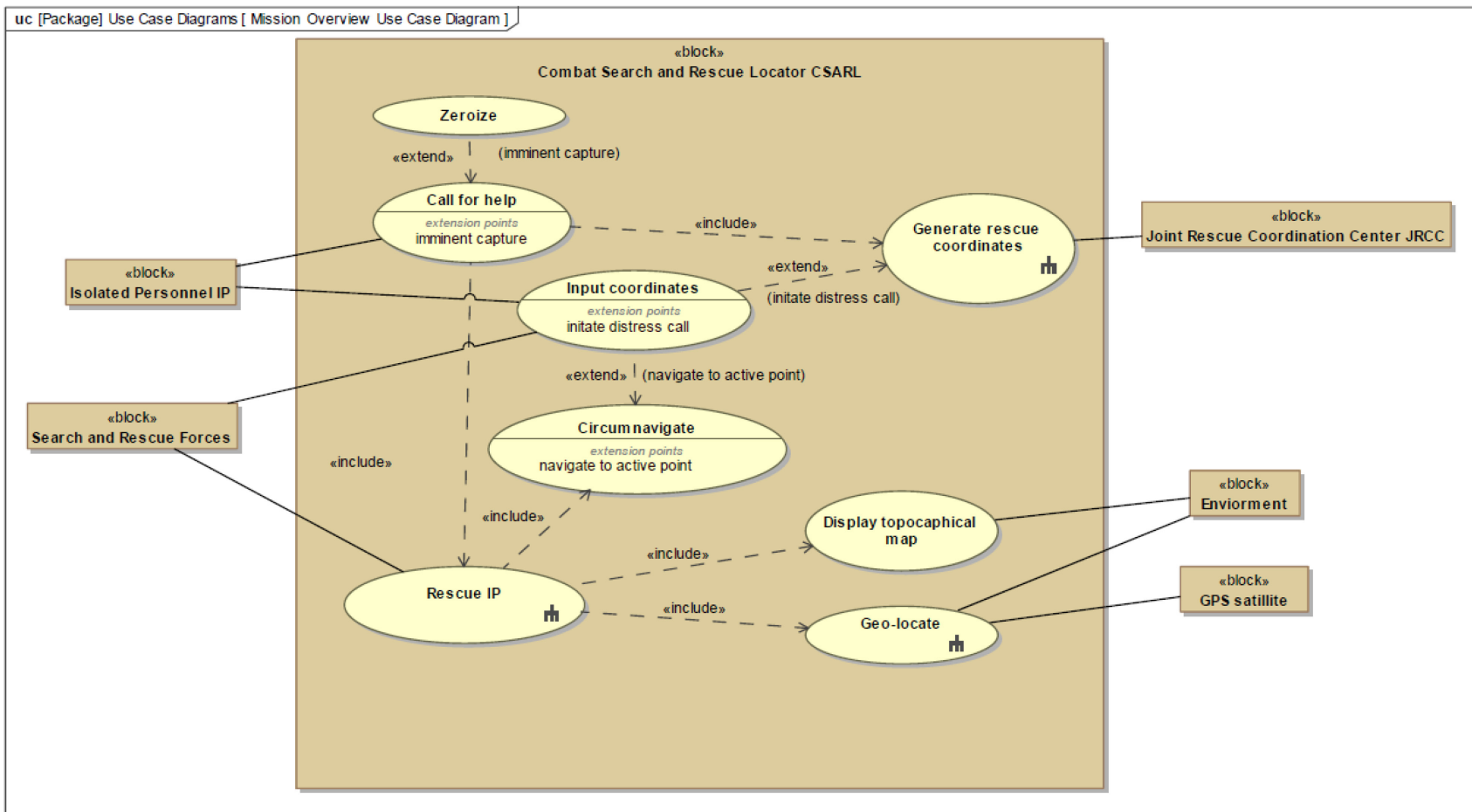


Figure 13. SysML Use Case Diagram of the Combat Search and Rescue Locator (CSARL) System used in the Case Study.

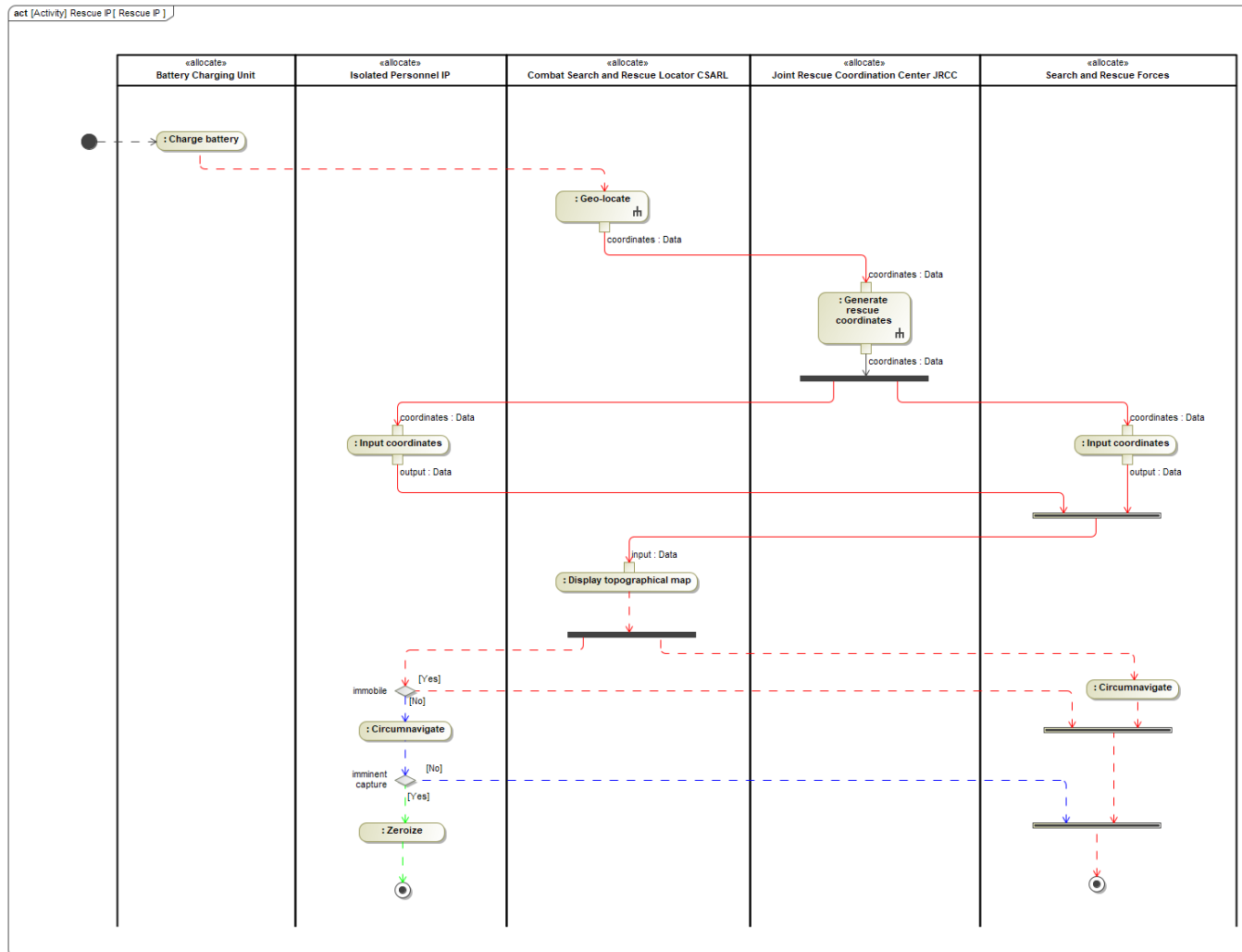


Figure 14. SysML Activity Diagram of the Combat Search and Rescue Locator (CSARL) System Used in the Case Study with Identified Path Flows.

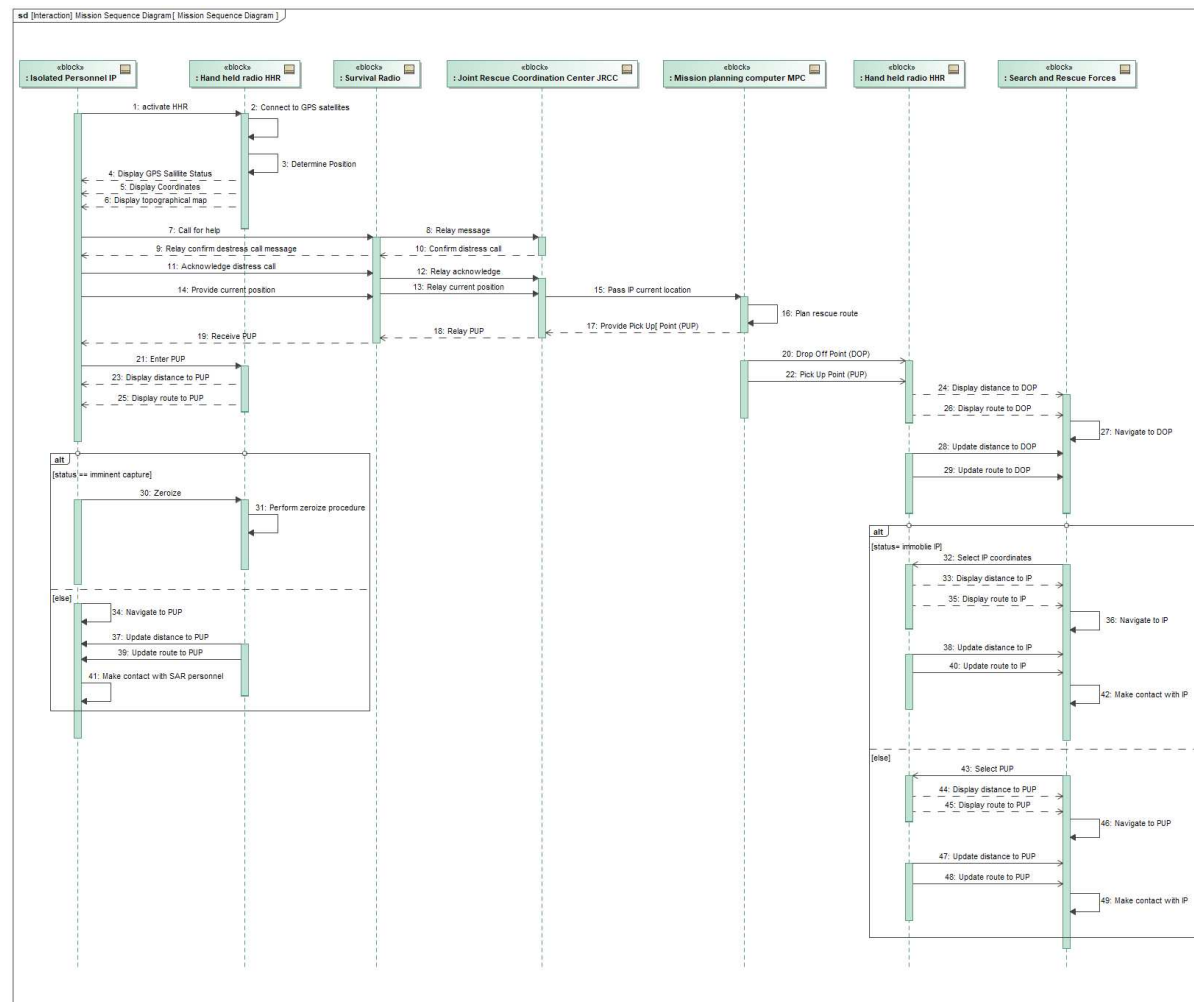


Figure 15. SysML Sequence Diagram of the Combat Search and Rescue Locator (CSARL) System used in the Case Study.

The interaction between the SUT and people engaged in the process (actors) [40] is documented in an activity diagram (Figure 14). The dialogs explain the actor's inputs or actions and the system interactions that are created in response [40]. For our CSARL example, the primary or essential flow is illustrated by the red path in Figure 14, which diagrams the situation where an aircraft experiences a critical mechanical failure, and the aircrew must eject.

The basic flow designates the aircrew as being injured and immobile and can be summarized as:

1. Call for help: The aircrew, now designated as isolated personnel (IP), initiates a call for help and provides secured GPS coordinates.
2. Generate Rescue Coordinates: The Joint Rescue Coordination Center (JRCC) authenticates the IP, relays the status and location information, and directs the search and rescue (SAR) forces to the IP's location.
3. Input coordinates: The SAR forces team enters a pre-designated point(s) where the IP has indicated their location.
4. Circumnavigate: The SAR forces circumnavigate to the designated IP coordinates for rescue and extraction.

In our CSARL case study, we have two alternate flows that can be executed, one by the aircrew personnel or IP and the other by the SAR forces. The first alternate flow designates the aircrew as able-bodied and able to proceed to a designated pick-up point (PUP). The blue path in Figure 14 illustrates alternate flow 1. The scenario is described as the JRCC directing both the IP and the SAR forces to a designated PUP coordinate where both parties circumnavigate for rescue and extraction. The second alternate flow is the case of imminent capture of the aircrew, where

they must perform a zeroized procedure that renders the handheld radio (HHR) inoperative. This is shown as the green path in Figure 14.

Once all flows have been identified in the SysML activity diagram, a table of all possible flows is created by applying the all-paths and all-configurations coverage criteria to generate all possible combinations using the grey box test scenario generation process and the basic flow and each alternate flow. For our example, Table 5 lists all the generated possible flows and configurations. Figure 16 illustrates the grey box test scenario generation process by using scenario 1 from Table 5 to generate the test procedure that outlines the steps required to execute the SAR team circumnavigation of the IP test case.

Table 5. All Configurations of Flows are Arranged into Various Scenarios

| <b>Document-Centric<br/>Scenario Name</b>                   | <b>Model-Centric<br/>Scenario Name</b>               | <b>Starting<br/>Flow</b> | <b>Alternate<br/>Flow</b> | <b>Exceptions<br/>Flow</b> |
|-------------------------------------------------------------|------------------------------------------------------|--------------------------|---------------------------|----------------------------|
| Scenario 1<br>Rescue Team Navigation<br>and Recovery        | Scenario 1<br>SAR Team Circumnavigate<br>to IP       | Basic flow               |                           |                            |
| Scenario 2<br>Isolated Personnel<br>Navigation and Recovery | Scenario 2<br>IP & SAR Team<br>Circumnavigate to PUP | Basic flow               | Alternate flow 1          |                            |
| Scenario 3<br>Zeroize HHR                                   | Scenario 3<br>Perform Zeroize Procedure              | Basic flow               |                           | Alternate flow 2           |

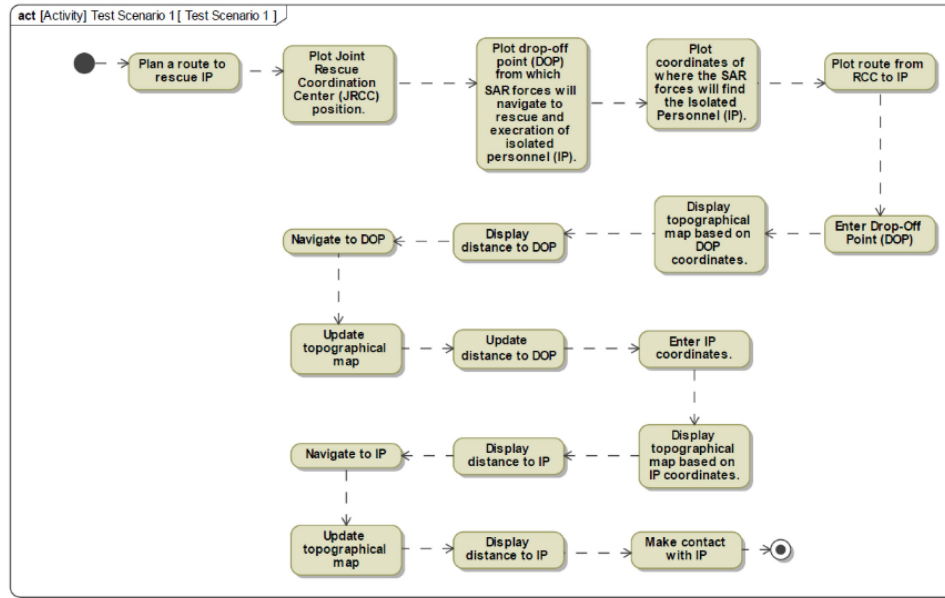


Figure 16. Activity Diagram of Scenario 1 Illustrating the Test Procedure that Outlines the Steps Required to Execute the SAR Team Circumnavigate to IP Test Case.

The case study illustrates that employing an MBSE framework in tandem with specific grey box testing techniques makes it possible to construct test scenarios efficiently. These generated test scenarios can then be seamlessly integrated into comprehensive test plans aimed at evaluating a system's capabilities [22]. The case study also shows the SysML model can effectively fulfill the need for well-defined documents outlining test scenarios tailored to the flight test objectives mandated by the acquisition process. Furthermore, the MDTD process can produce these essential documents with a level of reliability equal to the current document-centric method, yielding comparable elements crucial for successful testing [18]. The case study underscores the proposed approach as a viable method for generating test scenarios that contribute significantly to the test planning process.



### ***5.3.9 Results and Discussion***

This case study now allows a comparison between document-centric and model-centric processes.

#### ***5.3.9.1 Comparison of Processes Using Coverage Criteria***

The results of the model-based test scenario-generating method show that this method is effective in creating detailed test scenarios for use in the generation of detailed test plan documents (DTP)<sup>1</sup>. Comparing the results between the document-centric and model-centric processes reveals that the model-based method generates a comparable number of test scenarios as the document-centric process, as illustrated by contrasting the contents of Table 5 with the table of contents from the DTP illustrated in Figure 15. Similar outputs between the model-based test scenario and document-centric approaches provide confidence that both processes can generate similar documentation for testing SUTs.

Comparing the subheadings from the table of contents in the training DTP shown in Figure 12 with the steps outlined in the activity diagram in Figure 16, it is evident the model-based test scenario-generating method produces steps and content similar to the document-centric process. All the required steps from the training DTP have been accounted for and sufficient detail is presented in the SysML activity diagram (Figure 14) to generate a detailed DTP. As illustrated in Table 6, there is a simple and defensible mapping between the DTP steps generated from each process. Although the number of DTP steps and their level of detail is higher in the results of the

---

<sup>1</sup> Because the federal acquisition regulations (FAR) process is dictated by law, any new flight test process must create the equivalent documents for review and signature approval process required by the federal acquisition regulations, including the DTPs that are the output of this process.

model-centric process, Table 6 illustrates that the completeness and coverage of the model-centric test development process can be validated relative to the document-centric test development process.

Table 6. Scenario 1 Comparison of the Document-Centric DTP Steps with the Model-Centric Process-Generated Steps Derived from the SysML Activity Diagram.

| <b>Document-centric process</b>                        | <b>Model-centric Process</b>                       | <b>Remarks</b>                                                                                                                               |
|--------------------------------------------------------|----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Scenario 1: Rescue Team Navigation and Recovery</b> | <b>Scenario 1: SAR Team Circumnavigate to IP</b>   |                                                                                                                                              |
| Build RCC & IP Waypoints                               | Plan a route to rescue IP                          | Both processes accomplish the same task.                                                                                                     |
| Plot Rescue Coordination Center (JRCC)                 | Plot JRCC position                                 | Both processes accomplish the same task.                                                                                                     |
| Plot Isolated Personnel (IP) Waypoint                  | Plot coordinates where SAR forces will find IP     | Both processes accomplish the same task.                                                                                                     |
| Plot Dry Run route from RCC to IP                      | Plot Route from RCC to IP                          | Both processes accomplish the same task.                                                                                                     |
| Plot Drop-Off Point (DOP)                              | Plot DOP for SAR forces                            | Both processes accomplish the same task.                                                                                                     |
| Build RCC to DOP Route                                 | Plot Route from RCC to IP                          | The model-centric process combines all the document-centric steps into one phase.                                                            |
| Create a Route from RCC to DOP.                        |                                                    |                                                                                                                                              |
| Create a Route from DOP to IP.                         |                                                    |                                                                                                                                              |
| Download Route from MPC to HHR                         | Enter DOP                                          | Both processes accomplish the same task.                                                                                                     |
| Select RCC to IP Route                                 | Display Topographical Map based on DOP coordinates | Both processes accomplish the same task.                                                                                                     |
| Select Drop Off Point (DOP) Route                      | Display distance to DOP                            | The document-centric process combines all the model-centric steps into one phase. The detailed steps are broken down into minor subheadings. |
|                                                        | Navigate to DOP                                    |                                                                                                                                              |
|                                                        | Update Topographical map update distance to DOP    |                                                                                                                                              |
| Navigate to IP                                         | Enter IP coordinates                               | Both processes accomplish the same task.                                                                                                     |
|                                                        | Display Topographical Map based on IP coordinates  | The document-centric process combines all the model-centric steps into one phase. The detailed steps are broken down into minor subheadings. |
|                                                        | Display distance to IP                             |                                                                                                                                              |
|                                                        | Navigate to IP                                     |                                                                                                                                              |
|                                                        | Update Topographical map update distance to IP     |                                                                                                                                              |
| Rescue IP                                              | Make Contact IP                                    | Both processes accomplish the same task.                                                                                                     |

### ***5.3.9.2 Using Design-centric Models for Test Planning***

Although the developed testing plans using the document-centric and model-centric approaches define the same testing tasks, the model-driven process is not fault-free. Because the MBSE model developed to represent CSARL was built from a design perspective, it does not represent every connection that would be important from a testing and evaluation point of view.

For example, Figure 14 Illustrates two alternate flows: 1) test scenario 2: IP & SAR Team Circumnavigate to PUP, and 2) test scenario 3: Perform Zeroize Procedure. Focusing on test scenario 3, Figure 15 confirms that the alternate flow exists but does not provide details about the zeroing process. A comparison between the subheadings in the document-centric training DTP (Figure 12) and the steps in the model-centric process (Figure 15) demonstrates that the SysML activity diagram lacks sufficient detail to generate a detailed DTP for this case. A SysML activity diagram (Figure 17, which outlines the steps required to perform the zeroing process) is contained in the model, but this activity diagram is not explicitly referenced in the activity diagram of the CSARL system (Figure 14).

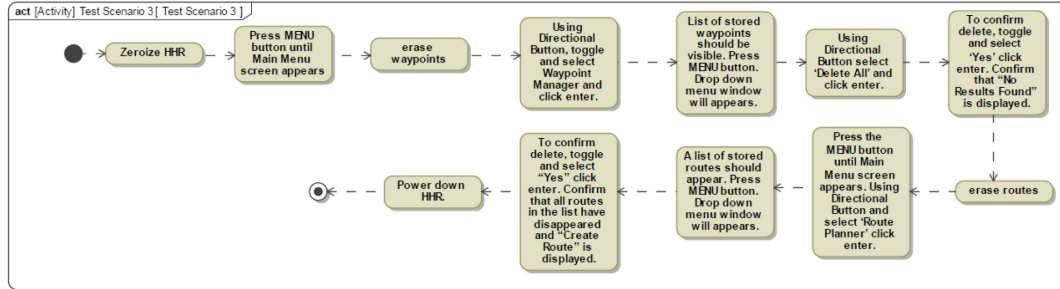


Figure 17. This Activity Diagram of Test Scenario 3 Illustrates the Test Procedure, which Outlines the Steps Required to Execute the “Perform Zeroize Procedure.”

The explicit reference for the zeroing process is shown in Figure 17 should be shown as a fork in the zeroize task contained in the Figure 14 activity diagram. The model designer creates links in models. In the case study, the model designer approached the model design from the development perspective without tester input.

The lack of tester input and review of the model leads to a lack of connectivity between the activity diagrams. Testers with SUT and MBSE software knowledge must provide oversight and input when creating the models. These testers serve as a quality control check to verify that all information required to produce quality test plans is available and documented in the model. Although the model-based approach can potentially eliminate the need for multiple SMEs and meetings to develop detailed test plans, the process must create written DTP documents controlled for quality through something like the traditional, expert-driven signature process.

### 5.3.10 Conclusions

The CSARL case study example shows that the novel, innovative grey box test scenario generation process can create the required documents at the same fidelity as the document-centric process [23] [18]. The case study shows the potential for efficient and valid use of SysML models for test plan design with MBSE and MDTD [35]. MBSE and grey box testing realize the promise

of digital transformation and constitute a stride toward modernizing testing methodologies to enhance system evaluation and validation [22] [39]. The CSARL case study exemplifies how adopting the MBSE method aids in formulating test scenarios and can contribute to comprehensive test plans for assessing a system's capabilities [39]. This enhancement can assist in the expedited development of test scenarios [18] [26] by being responsive to mission changes and technology changes that usually occur during a SUT acquisition life cycle. Embracing the paradigm shift toward MBSE in flight test planning provides progress toward successfully implementing DoD's Defense Digital Engineering Strategy [30] [33].

This exercise also provides evidence to support the idea that MBSE and MDTD do not replace the human element in test planning since the output quality depends on the test relevance and completeness of the model, which must be checked by capable personnel. This insight empowers testers to engage earlier in the architectural and design aspects of the model development [17]. Implementing MBSE and the concurrent generation of a testing and operational evaluation model allows the tester to embed testing information in the model of a SUT. Flight test planning and operational test planning can then occur without the physical asset, allowing test development earlier in the acquisition life cycle. Moreover, the creation of a digital test model leads to a digital test artifact that can be incorporated into a larger life-cycle model to capture the importance and value of flight and operational testing [35].

#### ***5.4 Remaining Tasks Associated with Research Question 1***

Research question 1 contained two other tasks not addressed in the publication. The tasks were to identify the metrics used in controlled software experimentation and examine which metrics used in software testing are suitable for implementation in flight testing.

### *5.4.1 Software Testing Metrics*

Organizations use metrics to obtain information to help improve productivity and reduce errors, products, and services [42]. Using metrics “throughout the testing effort leads to identifying risks and problems early in the test process, minimizing the impact of problems” [43]. Thus, metrics provide an avenue for identifying and tracking problems with a SUT’s development. Both the software and flight test community use metrics to effectively control costs and time, implement quality improvements, and reduce risk [43].

The metrics used by the software testing community fall into three categories: manual, production, and automated, but flight test is only concerned with those metrics that fall within the manual category and include [42]:

- Defect Acceptance (DA): This metric determines the number of valid defects the testing team identified during execution. To get a better picture, the value of this metric can be compared with previous releases.
- Defect Rejection (DR): This metric determines the number of defects rejected during execution. It gives the percentage of invalid defects the testing team has opened, which one can control, if necessary, in the future.
- Test Execution Productivity (TEP): This metric gives the test cases execution productivity, which, on further analysis, can provide conclusive results.
- Test Efficiency (TE): This metric determines the efficiency of the testing team in identifying defects. It also indicates which defects were missed during the testing phase and migrated to the next phase.

#### ***5.4.2 Metrics Suitable for Transfer to Flight Test***

The flight test community has developed its own set of metrics like those used by software testing and provides the same information as those described in task 4. Thus, no significant findings were found, and there were no major differences between what and how the software and flight test communities generate and report metrics.

Despite the goal of both the software and flight-testing communities to discover correct errors and deficiencies, the two societies differ in how this task is accomplished. The software testing community uses beta testing and incremental rollout of capabilities to collect bugs and errors. These issues are then documented and tracked for resolution before mass deployment to the targeted population or remain unresolved if the problem does not impact the overall function of the SUT or is a minor inconvenience to the user. Any issues found do not affect life or property and only reduce or stop the operation of a SUT. Therefore, software testing is only concerned with operation and satisfying client requirements.

Flight testing deals with one-of-a-kind or limited production of SUTs; thus, great care is taken not to damage or destroy an asset. Therefore, issues and problems discovered with a SUT are documented using a well-established process known as the Joint Deficiency Reporting System (JDRS) and are detailed in the Air Force's technical order (T.O.) 00-35D-34 [44]. The USAF Deficiency Reporting, Investigation, and Resolution technical manual "provides the Air Force with a means of identifying deficiencies, resolving those deficiencies within the bounds of program resources and the appropriate acceptance of risk for those deficiencies that cannot be resolved promptly" [44]. Therefore, no significant findings were found, and the flight test community has an engrained process for dealing with discovered problems and issues.

### ***5.5 Addressing Research Question 1***

After performing the above research tasks, I am equipped to address Research Question 1, which is restated as follows: *What software testing techniques are translatable to flight testing?*

This review has determined that various testing methods and techniques from multiple software practices could be adapted for integration into flight testing, thereby enabling the development of innovative testing methodologies and approaches for flight SUTs. These methods are presented in Table 4. A subset of these techniques was operationalized with the CSARL case study, illustrating the implementation of a grey box approach using the Model-Based Testing and Development (MDTD) process within a Model-Based Systems Engineering (MBSE) framework.

The research findings presented in this chapter have documented the testing methodologies presently employed in both software and flight testing. The research has summarized a collection of techniques originating from software testing engineering, but which have yet to find application in flight testing and are absent from the existing body of published flight-testing literature. The outcome of this review is an identification of numerous model-based testing procedures and concepts developed in software (such as state-model testing, class-diagram testing, thread-based testing, and use-based testing, all falling under the category of MBT and MDTD) that could contribute to model-based flight testing. These assessments support the idea that MBSE-enabled T&E will drive innovations in model-based testing as applied to flight systems.



## **CHAPTER 6 – A CASE STUDY-BASED ASSESSMENT OF A MODEL-DRIVEN TESTING METHODOLOGY AND METRICS OF MODEL REUSE**

### ***6.1 Introduction***

This chapter addresses Research Question 2, which is restated as follows: *What are the costs and benefits of the techniques associated with Research Question 1 when implemented in an MBSE construct?*

To answer this question, this chapter presents one baseline and two MBSE test case studies in which the MDTD is assessed for its cost and benefits regarding novel model and artifact reusability metrics.

### ***6.2 Publication Details***

The following discussion is based on my second publication, submitted to the International Test and Evaluation Association (ITEA) Journal of Test and Evaluation. The abstract was selected for presentation in the technical poster paper session of the annual International Test and Evaluation Symposium in Huntsville, Alabama, scheduled for November 2024. The annual ITEA Symposium is where organizations and professionals collaborate to advance and exchange technical, programmatic, and acquisition information among the T&E community. This work is fully replicated in this dissertation, with adjustments to conform to dissertation style guidelines. The publication's references have been integrated into the dissertation's reference section, though the content remains unchanged. The following discussion focuses on model reusability and metrics for measuring artifact reusability between models.

### ***6.2.1 Abstract***

The Department of Defense (DoD) Test and Evaluation (T&E) community has fully embraced digital engineering, as defined in the 2018 Digital Engineering Strategy, motivating the ongoing development and adoption of model-based testing methodologies. This article expands upon existing grey box model-driven test design (MDTD) approaches by leveraging model-based systems engineering (MBSE) artifacts to generate flight test planning models and documents. A baseline model of a system under test (SUT) and two additional system case studies are used to assess a Model-Driven Test Design (MDTD) process. The paper illustrates the method's applicability to these case studies, assesses the benefits of MDTD by applying novel metrics of model element reuse, and discusses the relevance to operational flight testing. This approach is novel within the flight-testing community, as it is the first implementation of MDTD in USAF operational testing applications. Whereas previous studies have explored SysML model reuse in small-scale problems or product families, MBSE model management for operational tests at flight-system scale and assessment of reuse in the T&E phase of the SE lifecycle are unresearched to date. This methodology and the case studies will be of particular interest to those involved in developing, executing, and reporting on flight test plans in the context of the DoD Digital Engineering transformation.

### ***6.2.2 Introduction***

Within the Department of Defense (DoD), flight test plans are developed to "communicate technical details and logistics for executing and reporting the outcomes of flight, ground, and laboratory tests on air vehicles, subsystems, and components" [16]. They establish a connection between test points and specified requirements, confirming that the SUT aligns with system specifications and delivers the intended performance in the proposed operational environment.

These plans encompass all aspects of test execution, including allocating critical test infrastructure and resources, proper data collection, and the procedures for post-flight analysis. Test plans capture and communicate a broad set of derived requirements that must be met for the successful execution of flight tests and are an integral aspect of the DoD acquisition process [3]. The conclusions drawn from the post-flight data analysis are then incorporated into reports to communicate and validate the SUT's flight test performance [45].

Within existing DoD acquisition processes, flight testing is an essential component of a sequential, document-centric, and resource-intensive process that often does not include early flight test planning. This sequential acquisition process commonly leads to prolonged cycle times, difficulties assimilating warfighter insights and feedback, an inability to swiftly integrate emerging technologies [28], and poorly justified or uncertain decision-making [2]. The existing flight test planning procedure is a meticulous, documentation-focused process that demands extensive configuration management efforts to ensure the currency of essential documents relative to the SUT. Considerations for flight testing most often arise late in the acquisition process, resulting in program delays due to test planning, instrumentation, test operations, and analysis. Generally, the traditional flight test planning activity is driven by consensus and expertise, which consumes significant man-hours and costs.

Within the context of the DoD's drive towards digital transformation, this research seeks to develop a model-based testing (MBT) methodology using model-based systems engineering (MBSE) concepts to enable improved integration of flight test planning with the digital engineering-driven acquisition process. This paper will expand on existing model-based grey box testing processes [28]. We develop and demonstrate a flight-test-specific MBT methodology using a set of digital engineering models that represent the product of a DoD MBSE-enabled product

development process. These models are scrutinized to assess their adequacy in providing the information necessary for test planning. The MBT process identifies missing parts or information by reviewing the native digital template and the resulting draft test plan. The proposed MBT process seeks to allow for the re-population and re-generation of a new version of the test plan when new information, updates, or new technology is inserted into the model. These assertions are tested through an assessment of the MBT methodology's ability to generate model-driven test plan artifacts that are updateable, consistent, and adaptable across acquisition programs [28].

### ***6.2.3 Background***

MBT is a subset of the emerging domain of MBSE [46]. The MBSE methodology comprises a set of interconnected processes, methods, and tools operating within a "model-based" or "model-driven" framework [34]. MBSE methodologies generate digital models that serve as abstract representations of reality [36]. These models include elements that rely on the power of computer models to capture, recall, and relate associated data. These elements can describe various aspects of a physical SUT's desired structure and behavior, business processes, and requirements. In practice, the set of interconnected state machine diagrams, use-case diagrams, sequence diagrams, and mathematical models enable testers to create independent test cases, scenarios, and plans abstracted from specific design implementations [47] [38]. Under the MBT paradigm, the model should precisely define test parameters and automatically generate detailed test plans for test cases. This enables automated and model-based testing, investigation, and optimization, lessening the dependence on physical assets. An MBSE testing model enables the parametric optimization of various test parameters, enabling agile or even evergreen reuse without rewriting the test plan when new requirements or changes to the SUT are introduced. This contrasts with the

current document-centric process, which requires manual revisions each time the SUT or its requirements change.

A component of MBT is Model-Driven Test Design (MDTD), which seeks to develop a comprehensive understanding of relationships and dependencies that govern test design, facilitating a thorough assessment of the testing implications of changes made at any stage of the product's design cycle [48]. MDTD significantly enhances the overall testing development process [48]. The MDTD process allows a subset of experienced testers to handle the high-level, statistical, and defensibility aspects of test design and development. In contrast, other testers plan the details of Validation and Verification (V&V) criteria, testing operations, and safety. This abstraction allows for the early involvement of designers responsible for the test planning process to incorporate testing requirements into the system design [49]. Ideally, MDTD empowers the operators and test team members to influence and contribute expertise in shaping and refining different aspects of a physical SUT's intended structure and behavior.

The DoD employs Digital Engineering (DE) methods and tools to transform how data is used to plan, design, develop, test, deploy, and sustain new systems and mission support capabilities [50]. While DE adoption is increasing, current programs face challenges due to existing infrastructure, processes, and workforce limitations. MBSE for T&E applies a digital approach to support test planning, execution, and the analysis of results while integrating model-based data as needed [50]. MBSE allows engineering teams to operate and handle the complexity of modern systems within a digital environment [50]. This digital environment promotes interoperability, helps identify conflicting requirements or technical gaps, and ensures configuration management as system designs evolve [50].

The "MBSE for T&E" roadmap presented in the "Positioning Test and Evaluation for the Digital Paradigm" [50] is a structured shift from a paper-based, siloed acquisition process to a data-centric T&E acquisition lifecycle as part of the DE transformation [50]. This roadmap is divided into three phases. Phase 1, Model-Based T&E Planning and Control, replaces traditional paper-based artifacts with digital ones, including descriptive and execution models. Phase 2, Dynamic T&E Planning and Execution, builds on the Phase 1 framework by incorporating quantitative methods that allow dynamic updates to test plans. Finally, Phase 3, Coupled Mission, System, and T&E Decision Making, integrates the Phase 2 framework with mission and systems engineering models [50].

#### ***6.2.4 Methods***

To assess the practical costs and benefits of MDTD, this study seeks to compare the processes by which test plan elements are used under current DoD guidance and the processes by which the same elements can be developed and reused under an MDTD test plan generation process. This section presents the characteristics of the existing test planning process, the elements that feed into the proposed MDTD test planning process, and the methods by which they will be compared. The comparison will be made using two test cases and one baseline case. AFOTEC's test plan generation process was used to develop the test plan elements using a document-centric and model-centric approach. Finally, the details of the comparison methods are presented as follows.

##### ***6.2.4.1 Baseline Operational Test & Evaluation Test Planning Process***

DOT&E requires proper T&E execution and the accomplishment of operational test and evaluation (OT&E) before an acquisition program enters full-rate production and reaches full operational capability. DOT&E accomplishes oversight by approving the adequacy of test plans

developed, written, and executed by AFOTEC personnel with assistance from DOT&E office personnel.

The OT&E Construct is the Air Force's framework for conducting an operational test. It identifies two elements of an operational test and their relationship to each other [45]:

- Operational Effectiveness measures the system's overall capability to accomplish a mission when used by representative personnel in the expected or planned operational environment while considering organization, doctrine, supportability, survivability, vulnerability, and threat.
- Operational Suitability measures the degree to which a system can be satisfactorily used in the field while considering its availability, compatibility, transportability, interoperability, reliability, wartime usage rates, maintainability, safety, human factors, manpower supportability, logistics supportability, natural environmental effects, and impacts, documentation, and training requirements.

These two elements of the operational test (Effectiveness & Suitability) are assessed by critical operational issues (COI), which address a vital mission element or operational objective to determine the system's overall capability to support mission accomplishment. These COIs are assigned measures designed to convey information about a system's effectiveness, capabilities, functions, properties, or structure. COIs are usually categorized as measures of effectiveness (MOE), measures of performance (MOP), or measures of suitability (MOS) [51]. These MOEs, MOPs, and MOSs are the criteria used to assess system behavior, capability, or operational environment changes. They measure the end state, achieve an objective, or create an effect. An MOE is designed to assess how the SUT can accomplish mission objectives and achieve desired mission effects. A MOP measures a system's capabilities, while a MOS measures the ability to

support the SUT in its intended operational environment [52]. Furthermore, each measure (MOS and MOP) is then composed of data elements, defined as any measurement or information used to determine the value of a measure. Data elements may be quantitative, qualitative, subjective, or objective. Figure 18 illustrates the hierarchy of these measures outlined in red and how they fall within the overall AFOTEC OT&E construct.

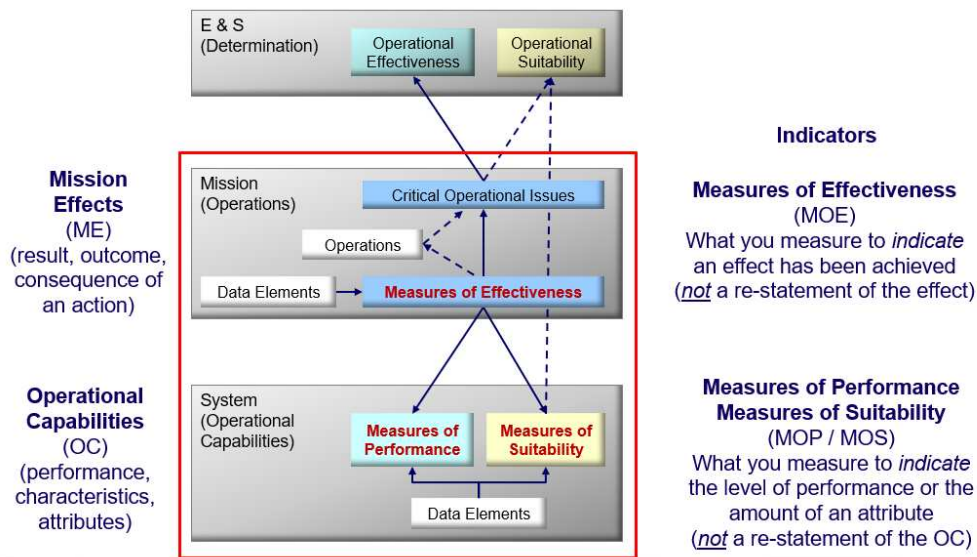


Figure 18. Measure Hierarchies and the Overall AFOTEC OT&E Construct

AFOTEC uses a well-established test planning process outlined in the relevant guides [45] and [51]. The flight-testing community has not demonstrated this process in a digital engineering construct, nor have acquisition systems been developed using an MBSE approach. The methods proposed in this paper define how this process may be adapted to enable an MDTD test planning process for future digital engineering-enabled test articles.

The current document-centric test plan development process encourages test planners to review and research previous test plans as a starting point for drafting new plans. This approach



relies heavily on "copy and paste" reuse to develop test plan elements such as COIs, MOEs, MOPs, and MOS. The challenges associated with this document-centric reuse are that COIs, MOEs, MOPs, and MOS are often interrelated, and the relationships between them are not usually explicitly captured or crosslinked in the test plan document. Their relationships are such that small changes to a single MOE realize significant changes to multiple MOPs or MOSs, which appear numerous times throughout the document and must be manually updated. As test plans evolve, these elements often change, requiring planners to update them repeatedly across different sections. This repetitive process consumes significant time, manpower, and resources. Additionally, the document-centric approach lacks an effective reuse mechanism due to personnel turnover, and team committees and stakeholders need to reconvene and debate changes. In general, although the reuse of test plans is possible within a document-based systems engineering construct, the utility of reused test plan documents is, in practice, very low, and every test plan is functionally independent.

#### ***6.2.4.2 Proposed Model-Based AFOTEC Testing Methodology***

Test planning is identifying and communicating the conditions to be implemented during testing. Test plans are the systems engineering artifact that communicates successful and acceptable implementation of product requirements [8]. Test plans are devised to determine the measures required to confirm the implementation of product requirements that align with the warfighter's mission objectives and that provide insights into the system's structure and usability in the field [53]. Using an MBSE-enabled process, testing engineers can generate test plans by building test models during the design and development of the product and associated models. This approach facilitates the modeling of the testing process parallel to the design and guides all users on the testing requirements and methodology.

A pragmatic approach to formulating a test plan from a given model involves furnishing the model with testing elements to be utilized in developing the test plan. These elements should contain the critical measures (as illustrated in Figure 18) and operational details needed to create an operational test plan. Developing and implementing a testing model based on the OT&E construct within an MBSE-enabled MDTD enables the creation of test plan models.

#### ***6.2.4.3 Model Reuse Assessment***

In this study, we have developed a set of case studies that can be used to derive examples of MBSE-enabled test plan models. We then determine the degree of reuse enabled by the proposed MBSE-enabled MDTD process. Although many benefits have been asserted in literature for an MBSE-enabled MDTD process, model reuse is stated consistently to be a key benefit of MBSE relative to the conventional document-centric test planning process [35] [46]. In the context of the continuous testing activity that is the responsibility of AFOTEC, model reuse is a crucial characteristic of improving the efficiency and effectiveness of test plan development under the Digital Engineering construct. Sharing information and automating administrative tasks save time and facilitate process reuse [4].

Since MBSE is still in the early stages of adoption within the DoD and the aerospace industry, a clear definition and ontology of SysML model reuse is required. In this study, we adopt a definition of reuse from software engineering [54]. In the software engineering community, reuse often involves a "copy & paste" method, where specific artifacts are copied and adapted to meet the current project's needs [55]. This reuse extends the software artifact's functionality, enabling it to be integrated or repurposed for different applications [54], albeit with a system of systems emergent risks from the multi-proprietary and multi-generational nature of such reuse [56].

To achieve significant benefits, reuse must be systematic, and organizations implementing reuse programs must measure their strategies' effectiveness [57]. Reuse can be classified into three distinct categories [55]:

1. **Identical Subset Reuse:** This occurs when a subset of initial functions or elements are identical to a subset of the new functions or elements, requiring only minor changes such as renaming.
2. **Modified Subset Reuse:** This involves a subset of initial functions or elements that are modified versions of the new functions or elements. This category can be further divided into:
  - a. **Minor Modifications:** This involves minor changes, such as renaming or altering input/output values.
  - b. **Major Modifications:** Significant modifications to the content are required while maintaining the same structure.
3. **Creation of New Elements:** This involves the creation of entirely new functions or elements.

A standard metric used to measure systematic software reuse is Product Reuse Percent (PRP), which indicates how much of the product can be attributed to reuse. This metric is calculated using the amount of reused source instructions (RSI) and shipped source instructions (SSI). Reuse percent is calculated as follows [57]:

$$PRP = \frac{RSI}{RSI + SSI} * 100 \quad (1)$$

When the product has no reuse instructions,  $RSI=0$ , resulting in  $PRP=0$ . When  $RSI=SSI$ , all shipped instructions are the result of reuse,  $RSI=SSI$ , and  $PRP=1$ .

An alternate metric of software reuse is Reuse Value Added (RVA) [58]. This metric evaluates the potential benefits of consuming and producing reusable software and is based on a productivity index that determines the effectiveness of reuse across multiple products. The RVA is calculated as follows [58]:

$$RVA = \frac{(SSI + RSI) + SIRBO}{SSI} \quad (2)$$

Products that do not reuse code have  $RVA = 1$ , while an  $RVA = 2$  indicates that the product is twice as effective than if the product was created without reuse [58]. The total effectiveness of a development product is higher when RVA scores are higher.

Source Instructions Reused by Others (SIRBO) represent the total lines of code that new products reuse from existing ones. SIRBO measures the extent and success of reusing components from older products [58]. Organizations that develop successful, reusable components will have a high SIRBO, as the metric increases each time another product reuses the software. SIRBO is high in organizations that produce high-quality, well-documented, broadly applicable, and reusable components [58]. SIRBO is calculated by summing the code lines for all product parts that contribute to reuse.

$$SIRBO = [(\text{Number of Products}) \times \text{Lines of Code Reused}] \quad (3)$$

Substituting the SIRBO equation into the RVA calculation provides a simple formula using direct measurements of a product [58].

$$RVA = \frac{(SSI + RSI) + [(Number\ of\ Products) \times Lines\ of\ Code\ Reused]}{SSI} \quad (4)$$

Since our models contain modeling elements and not lines of code, we will redefine RSI as the number of reused elements (RSE) and SSI as the number of model elements (ME). Thus, we can define a new metric for model reusability, called Model Reuse percent (MR%, analogous to PRP), which is calculated as follows:

$$MR\% = \frac{RSE}{RSE + ME} * 100 \quad (5)$$

Moreover, since lines of code reused in the SIRBO equation are equivalent to reused elements, and the number of products is the same as the number of models, we can restate the RVA formula for use in modeling as follows:

$$RVA = \frac{(ME + RSE) + [(Number\ of\ Models) \times RSE]}{ME} \quad (6)$$

The RM% metric primarily assesses the reuse of artifacts between two consecutive generations of models. On the other hand, the RVA metric can evaluate reuse between consecutive model generations and between a baseline model and subsequent models that are multiple generations removed. Figure 19 illustrates the relationships between these two metrics and the models. Both the RM% and RVA metrics will be applied to evaluate the extent of reuse.

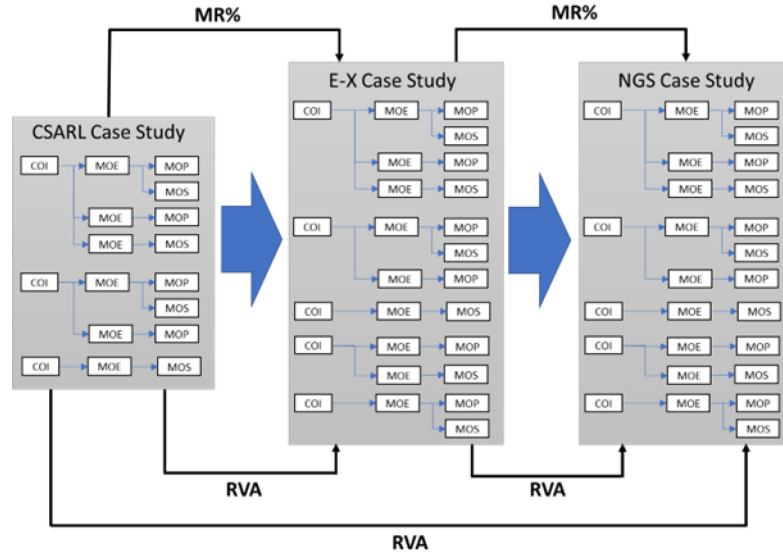


Figure 19. MR% and RVA Metrics as Applied Between the Generations of Case Study Models.

The proposed refined RVA metric offers a straightforward way to assess how effectively elements are reused from the baseline model through multiple generations of reuse in other models. A score above 1 signifies successful reuse, indicating that developers consistently reuse components with each new model iteration. This metric encourages developers to create high-quality, well-documented, widely applicable components. By doing so, it supports the creation of more reusable artifacts. Ultimately, this leads to improved efficiency and sustainability in the development process [58].

### 6.2.5 Case Studies Definition

Three case studies were developed and produced using SysML implemented in MBSE with the Dassault Systemes' Cameo Modeler™ software package to execute and assess the proposed process. The first case study model, the Combat Search and Rescue Locator (CSARL), was created from a design and development point of view. The model excluded participation and inputs from the testing community during its development. The second model, the E-X airborne multi-sensor

platform, is a revision and update of an example in the textbook “Effective Model-Based Systems Engineering” [35]. The third model, the next-generation sensor (NGS) system, is a generalized version of a “real-world” example model developed by the Air Force Material Command (AFMC) digital engineering program office. The NGS model teaches government and contractor personnel how to use and implement the weapon government reference architecture (weapons GRA) used to standardize the Air Force’s implementation of digital engineering into the DoD’s acquisition process. Details of these case studies and why they were chosen are presented in the following paragraphs.

The baseline case study focuses on AFOTEC’s Combat Search and Rescue Locator (CSARL) system. CSARL is a fictional system developed as a training case for AFOTEC flight test personnel learning to create test plans. It is well-documented, well-accepted in the testing community, and has a thorough test plan “answer key”. The CSARL system is notionally a portable hand-held Global Positioning System (GPS) receiver system offering enhanced digital moving maps and real-time navigational capabilities for downed aircrew and search and rescue (SAR) forces during combat and peacetime survival scenarios. Its purpose is to complement current GPS-integrated survival radios, enhancing existing survival and rescue navigation capabilities without relying on traditional map-and-compass techniques [59]. CSARL is an example system demonstrating the proposed methods for relevant flight-test test plan generation. Since the system is well-documented and all AFOTEC personnel are familiar with the application and model, it is a good benchmark for applying the grey box-inspired testing technique implemented in an MBSE environment to create a flight test plan without direct SME intervention. Moreover, it provides a means to compare outputs between the document-centric and model-centric processes.

The first case study uses a system described in the Effective Model-Based Systems Engineering textbook [35]. The textbook introduces the “E-X Airborne Sensor Platform” as a hypothetical system used to exemplify the development of various diagrams for defining and describing software-intensive aerospace systems using System Modeling Language (SysML). The E-X airborne multi-sensor platform gathers intelligence, surveillance, and reconnaissance (ISR) data. Operating in collaboration with ground stations and affiliated organizations, it facilitates the collection and dissemination of ISR products. E-X platform executes ISR missions and performs aerial imaging and sensing functions [35]. The E-X case study is a valuable example of a moderately complex system, utilizing SysML diagrams readily accessible in the textbook [35] and easily replicated. Like the CSARL case study, the E-X system includes comprehensive documentation, making it an ideal standard for applying grey box-inspired testing techniques within an MBSE environment. Additionally, it offers another podium for comparing outcomes between document-centric and model-centric processes.

The second case study uses an example model developed by the AFMC digital engineering program office called Next Generation Sensor Platform (NGS) System [60]. The NGS system is a notional airborne sensor platform that can be configured to use various sensors to aid the ISR collection mission. The NGS test case study provides a practical illustration of a moderately complex physical system, utilizing SysML diagrams to which the grey box-inspired testing techniques within an MBSE construct can be applied.

Table 7 summarizes the features of each case study model and visually demonstrates the increase in complexity across the test cases. The number of elements and diagrams increases as the test cases become more representative of real-world complex systems, outlining the characteristics and highlighting the correlation between system complexity and the number of



elements and diagrams. The selection of two case studies and one baseline case for this research aims to test the applicability to diverse systems with distinct missions and similar systems with identical missions but varying degrees of complexity.

Table 7. Summary Table of Case Study Model Characteristics

| <b>Model Characteristics</b> | <b>CSARL</b>                   | <b>E-X</b>                                                  | <b>NGS</b>                                                  |
|------------------------------|--------------------------------|-------------------------------------------------------------|-------------------------------------------------------------|
| <b>System Mission</b>        | <b>Search and Rescue (SAR)</b> | <b>Intelligence, Surveillance, and Reconnaissance (ISR)</b> | <b>Intelligence, Surveillance, and Reconnaissance (ISR)</b> |
| Number of Elements           |                                |                                                             |                                                             |
| Requirements                 | 26                             | 8                                                           | 51                                                          |
| Structure                    | 15                             | 35                                                          | 242                                                         |
| Behavior                     | 7                              | 63                                                          | 86                                                          |
| Number of SysML Diagrams     |                                |                                                             |                                                             |
| Activity                     | 11                             | 20                                                          | 39                                                          |
| Block                        | 23                             | 25                                                          | 54                                                          |
| Internal Block               | 0                              | 8                                                           | 15                                                          |
| Requirement                  | 1                              | 1                                                           | 14                                                          |
| Sequence                     | 1                              | 1                                                           | 4                                                           |
| State Machine                | 1                              | 1                                                           | 28                                                          |
| Use Case                     | 1                              | 2                                                           | 5                                                           |

### ***6.2.6 Applicability Assessment Methods***

The CSARL case study was used to create and develop the model-based AFOTEC testing model (Figure 20), which includes all the elements used in a test plan generation process, as outlined in the Operational Test & Evaluation Construct section. Each folder within the “AFOTEC Methodology” folder of the CSARL system was populated with details to meet the requirements of the AFOTEC test planning guide [45]. Thus, details of each MOE, MOP, and MOS linked to each COI were also populated with all the necessary specifics that each element requires to meet the AFOTEC test planning guide [45]. Figure 20 shows the details of the COI, MOE, MOP, and MOS subfolders contained within the AFOTEC Methodology folder for the CSARL case study. This methodology was then used to generate a sample test plan using the elements within the

AFOTEC test structure. The plan was compared with its document-centric counterpart and assessed for content and methodology.

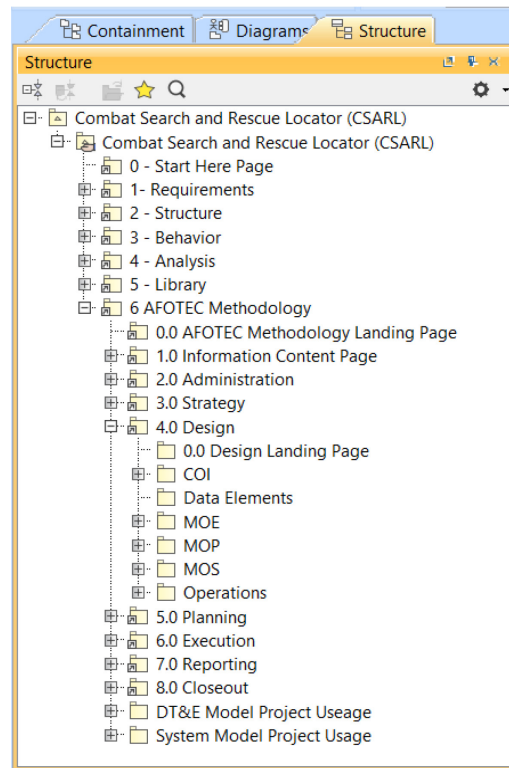


Figure 20. The CSARL Model Layout Incorporates the AFOTEC Test Model, Labeled “AFOTEC Methodology.”

Additionally, once the “AFOTEC Methodology” model was created and assessed to contain all the information needed to generate a preliminary draft test plan, the structure was adapted to the E-X airborne multi-sensor platform model. To match the requirements of the E-X test planning, the AFOTEC test plan model elements required varying levels of modification, from simple edits to change names and parameters to match the model, to significant changes that are required rewriting of needed information to provide detail and tailor the element for use relevant to the model functions or behaviors associated with the aspect. This level of adjustment was

expected since the CSARL and the E-X platform models are very different in scope, concept, and mission.

Once the elements within the “AFOTEC Methodology” model for the E-X platform model were modified, they were assessed for content to ensure that each MOE, MOP, and MOS was linked to its associated COI and populated with all the essential details required to meet the AFOTEC test planning guide [45]. Table 8 shows the details of the COI, MOE, MOP, and MOS subfolders contained within the AFOTEC Methodology Model for the E-X test case study.

Table 8. Name of Elements Contained Within the CSARL Model Structure Subfolders.

| Element Names for CSARL Test Case contained within the AFOTEC Methodology Model Structure |                                               |                                                                                       |                                                                                          |
|-------------------------------------------------------------------------------------------|-----------------------------------------------|---------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| COI Folder                                                                                | MOE Folder                                    | MOP Folder                                                                            | MOS Folder                                                                               |
| Can CSARL enable rescue team navigation to an isolated personnel location?                | MOE 1.1 Support of IP Location                | MOP 1.1.1 HHR stationary geo-location position                                        |                                                                                          |
|                                                                                           |                                               | MOP 1.1.2 The HHR stationary geo-location elevation                                   |                                                                                          |
|                                                                                           |                                               | MOP 1.1.3 HHT geo-position error accuracy                                             |                                                                                          |
| Can CSARL enable isolated personnel navigation to a rendezvous location?                  | MOE 2.1 Support of Rendezvous Points Location | MOP 2.1.1 HHR stationary geo-location position                                        |                                                                                          |
|                                                                                           |                                               | MOP 2.1.2 The HHR stationary geo-location elevation                                   |                                                                                          |
|                                                                                           |                                               | MOP 2.1.3 HHT geo-position error accuracy                                             |                                                                                          |
|                                                                                           |                                               | MOP 2.1.4 HHR circumnavigates around obstacles without adding additional waypoints    |                                                                                          |
|                                                                                           |                                               | MOP 2.1.5 HHR circumnavigates obstacles and still arrives at the designated objective |                                                                                          |
|                                                                                           |                                               | MOP 2.1.6 HHR detail for situational awareness                                        |                                                                                          |
|                                                                                           |                                               | MOP 2.1.7 HHR time to acquire and display geo-location                                |                                                                                          |
|                                                                                           | MOE 2.2 HHR Cleared Successfully              | MOP 2.2.1 HHR Cleared Successfully                                                    |                                                                                          |
| Does CSARL support worldwide search and rescue operations?                                | MOE 3.1 CSARL Available for Operations        |                                                                                       | MOS 3.1.1 Operational Availability (Ao)                                                  |
|                                                                                           |                                               |                                                                                       | MOS 3.1.2 Availability of MPC for mission planning                                       |
|                                                                                           |                                               |                                                                                       | MOS 3.1.3 HHR battery life, continuous navigation                                        |
|                                                                                           |                                               |                                                                                       | MOS 3.1.4 Mission Reliability (MR)                                                       |
|                                                                                           |                                               |                                                                                       | MOS 3.1.5 Mean Repair Time (MRT)                                                         |
|                                                                                           |                                               |                                                                                       | MOS 3.1.6 Usability of HHR map detail                                                    |
|                                                                                           |                                               |                                                                                       | MOS 3.1.7 Successful transfer of planned mission from the MPC to the HHR without failure |

Table 9. Name of Elements Contained Within the E-X Model Structure Subfolders.

| Element Names for E-X Platform Test Case contained within the AFOTEC Methodology Model Structure |                                                                                |                                                                                     |                                                                                          |
|--------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| COI Folder                                                                                       | MOE Folder                                                                     | MOP Folder                                                                          | MOS Folder                                                                               |
| Can the E-X Sensor Platform perform wide-area surveillance?                                      | MOE 1.1 Continuous detection, identification, and tracking of airborne targets | MOP 1.1.1 Targets Displayed as Single Airborne Tracks                               |                                                                                          |
|                                                                                                  |                                                                                | MOP 1.1.2 On-board Airborne tracks with a Track Quality of 7 better                 |                                                                                          |
|                                                                                                  |                                                                                | MOP 1.1.1 Targets Displayed as Single Airborne Tracks                               |                                                                                          |
|                                                                                                  | MOE 1.2 Continuous detection, identification, and tracking of ground targets   | MOP 1.2.1 Targets Displayed as Single Ground Tracks                                 | MOS 1.2.1 Operator Rating of Ground Track Performance                                    |
|                                                                                                  |                                                                                | MOP 1.2.2 On-board Ground tracks with a Track Quality of 7 better                   |                                                                                          |
|                                                                                                  | MOE 1.3 Continuous detection, identification, and tracking of maritime targets | MOP 1.3.1 Targets Displayed as Single Maritime Tracks                               | MOS 1.3.1 Operator Rating of Maritime Track Performance                                  |
|                                                                                                  |                                                                                | MOP 1.3.2 On-board Maritime tracks with a Track Quality of 7 better                 |                                                                                          |
|                                                                                                  | MOE 1.4 Continuous detection, identification, and tracking of ESM targets      | MOP 1.4.1 ESM Geo-location                                                          | MOS 1.4.1 Operator Rating of ESM Performance                                             |
|                                                                                                  |                                                                                | MOP 1.4.2 ESM Geo-location Error                                                    |                                                                                          |
| Can the E-X Platform collect signal data to support Combatant Commander taskings?                | MOE 2.1 Correct identification of targets for tactical picture                 | MOP 2.1.1 Friendly Air Vehicles Correctly Identified                                |                                                                                          |
|                                                                                                  |                                                                                | MOP 2.1.2 Enemy Air Vehicles Correctly Identified                                   |                                                                                          |
|                                                                                                  |                                                                                | MOP 2.1.3 Ground/Surface Targets Correctly Identified                               |                                                                                          |
|                                                                                                  | MOE 2.2 Maintain communication and distribution of the common tactical picture | MOP 2.2.1 Critical Information Exchange Requirements Met                            | MOS 2.2.1 Battlespace Participant Rating of E-X Cueing                                   |
|                                                                                                  |                                                                                | MOP 2.2.2 Track Reports Transmitted IAW Normal Link Operations with Track Quality 7 | MOS 2.2.2 Joint/Combined Participant Rating of E-X Contribution to the Battlespace       |
|                                                                                                  |                                                                                |                                                                                     | MOS 2.2.3 Mission Crew Rating of Situational Awareness                                   |
|                                                                                                  |                                                                                |                                                                                     | MOS 2.2.4 Operator Rating of Battle Management Task Efficiency                           |
| Can the E-X Platform support the exploitation and dissemination of ISR information to end users? | MOE 3.1 Generation Signal Products                                             | MOP 3.1.1 SIGINT report correctness                                                 | MOS 3.1.1 End user rating of intelligence, surveillance, and reconnaissance data utility |
|                                                                                                  |                                                                                | MOP 3.1.2 Number of electromagnetic compatibility issues                            | MOS 3.1.2 Mission manager's rating of ability to perform mission management functions    |
|                                                                                                  | MOE 3.2 Successful Transmission of Messages                                    | MOP 3.2.1 Percentage of imagery data successfully transmitted                       |                                                                                          |
|                                                                                                  |                                                                                | MOP 3.2.2 Time for imagery transmission via common data link                        |                                                                                          |
|                                                                                                  |                                                                                | MOP 3.2.3 Percentage of signals successfully transmitted                            |                                                                                          |
|                                                                                                  |                                                                                |                                                                                     |                                                                                          |
| Is the E-X Airborne Sensor Platform employable to support the TACS?                              | MOE 4.1 Generate Mission Plan                                                  |                                                                                     | MOS 4.1.1 Operator Rating of Mission Planning                                            |
|                                                                                                  |                                                                                |                                                                                     | MOS 4.1.2 Technician Rating of Mission Planning                                          |
|                                                                                                  | MOE 4.2 Conduct Post Mission Analysis                                          |                                                                                     | MOS 4.2.1 Operator Rating of Mission Debrief                                             |
|                                                                                                  |                                                                                |                                                                                     | MOS 4.2.2 Technician Rating of Mission Debrief                                           |
|                                                                                                  | MOE 4.3 Conduct Training Analysis                                              |                                                                                     | MOS 4.3.1 Operator Rating of Training                                                    |
|                                                                                                  |                                                                                |                                                                                     | MOS 4.3.2 - Technician Rating of Training                                                |
|                                                                                                  |                                                                                |                                                                                     | MOS 4.3.3 Operator Rating of Training Documentation                                      |
|                                                                                                  |                                                                                |                                                                                     | MOS 4.3.4 Technician Rating of Training Documentation                                    |
| Is the E-X Airborne Sensor Platform sustainable to support the TACS?                             | MOE 5.1 Aircraft generation and recovery                                       |                                                                                     | MOS 5.1.1 Operational Availability                                                       |
|                                                                                                  |                                                                                |                                                                                     | MOS 5.1.2 Ground Repair Time                                                             |
|                                                                                                  |                                                                                |                                                                                     | MOS 5.1.3 Maintainer Rating of System Maintainability                                    |
|                                                                                                  |                                                                                |                                                                                     | MOS 5.1.4 Mission System Time Between Unscheduled Maintenance                            |

Table 10. Name of Elements Contained Within the NGS Model Structure Subfolders.

| Element Names for NGS System Test Case contained within the AFOTEC Methodology Model Structure |                                                                                |                                                                                     |                                                                                          |
|------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------|-------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|
| MOP Folder                                                                                     | MOS Folder                                                                     | MOP Folder                                                                          | MOS Folder                                                                               |
| Can the NGS Sensor Platform perform wide-area surveillance?                                    | MOE 1.1 Continuous detection, identification, and tracking of airborne targets | MOP 1.1.1 Targets Displayed as Single Airborne Tracks                               | MOS 1.2.1 Operator Rating of Ground Track Performance                                    |
|                                                                                                |                                                                                | MOP 1.1.2 On-board Airborne tracks with a Track Quality of 7 better                 |                                                                                          |
|                                                                                                | MOE 1.2 Continuous detection, identification, and tracking of ground targets   | MOP 1.1.1 Targets Displayed as Single Airborne Tracks                               |                                                                                          |
|                                                                                                |                                                                                | MOP 1.2.1 Targets Displayed as Single Gound Tracks                                  |                                                                                          |
|                                                                                                | MOE 1.3 Continuous detection, identification, and tracking of maritime targets | MOP 1.2.2 On-board Ground tracks with a Track Quality of 7 better                   |                                                                                          |
|                                                                                                |                                                                                | MOP 1.3.1 Targets Displayed as Single Maritime Tracks                               | MOS 1.3.1 Operator Rating of Maritime Track Performance                                  |
|                                                                                                | MOE 1.4 Continuous detection, identification, and tracking of ESM targets      | MOP 1.3.2 On-board Maritime tracks with a Track Quality of 7 better                 |                                                                                          |
|                                                                                                |                                                                                | MOP 1.4.1 ESM Geo-location                                                          | MOS 1.4.1 Operator Rating of ESM Performance                                             |
| Can the NGS System collect signal data to support Combatant Commander taskings?                | MOE 2.1 Correct identification of targets for tactical picture                 | MOP 1.4.2 ESM Geo-location Error                                                    |                                                                                          |
|                                                                                                |                                                                                | MOP 2.1.1 Friendly Air Vehicles Correctly Identified                                |                                                                                          |
|                                                                                                |                                                                                | MOP 2.1.2 Enemy Air Vehicles Correctly Identified                                   |                                                                                          |
|                                                                                                | MOE 2.2 Maintain communication and distribution of the common tactical picture | MOP 2.1.3 Ground/Surface Targets Correctly Identified                               |                                                                                          |
|                                                                                                |                                                                                | MOP 2.2.1 Critical Information Exchange Requirements Met                            | MOS 2.2.1 Battlespace Participant Rating of NGS Cueing                                   |
|                                                                                                |                                                                                | MOP 2.2.2 Track Reports Transmitted IAW Normal Link Operations with Track Quality 7 | MOS 2.2.2 Joint/Combined Participant Rating of NGS Contribution to the Battlespace       |
|                                                                                                |                                                                                |                                                                                     | MOS 2.2.3 Mission Crew Rating of Situational Awareness                                   |
|                                                                                                |                                                                                |                                                                                     | MOS 2.2.4 Operator Rating of Battle Management Task Efficiency                           |
| Can the NGS System support exploiting and disseminating ISR information to end users?          | MOE 3.1 Generation Signal Products                                             |                                                                                     |                                                                                          |
|                                                                                                |                                                                                | MOP 3.1.1 SIGINT report correctness                                                 | MOS 3.1.1 End user rating of intelligence, surveillance, and reconnaissance data utility |
|                                                                                                | MOE 3.2 Successful Transmission of Messages                                    | MOP 3.1.2 Number of electromagnetic compatibility issues                            | MOS 3.1.2 Mission managers' rating of ability to perform mission management functions    |
|                                                                                                |                                                                                | MOP 3.2.1 Percentage of imagery data successfully transmitted                       |                                                                                          |
|                                                                                                |                                                                                | MOP 3.2.2 Time for imagery transmission via common data link                        |                                                                                          |
|                                                                                                |                                                                                | MOP 3.2.3 Percentage of signals successfully transmitted                            |                                                                                          |
| Is the NGS Airborne Sensor Platform employable to support the TACS?                            | MOE 4.1 Generate Mission Plan                                                  |                                                                                     | MOS 4.1.1 Operator Rating of Mission Planning                                            |
|                                                                                                | MOE 4.2 Conduct Post Mission Analysis                                          |                                                                                     | MOS 4.1.2 Technician Rating of Mission Planning                                          |
|                                                                                                |                                                                                |                                                                                     | MOS 4.2.1 Operator Rating of Mission Debrief                                             |
|                                                                                                | MOE 4.3 Conduct Training Analysis                                              |                                                                                     | MOS 4.2.2 Technician Rating of Mission Debrief                                           |
|                                                                                                |                                                                                |                                                                                     | MOS 4.3.1 Operator Rating of Training                                                    |
|                                                                                                |                                                                                |                                                                                     | MOS 4.3.2 Technician Rating of Training                                                  |
|                                                                                                |                                                                                |                                                                                     | MOS 4.3.3 Operator Rating of Training Documentation                                      |
|                                                                                                |                                                                                |                                                                                     | MOS 4.3.4 Technician Rating of Training Documentation                                    |
| Is the NGS Airborne Sensor Platform sustainable to support the TACS?                           | MOE 5.1 Aircraft generation and recovery                                       |                                                                                     | MOS 5.1.1 Operational Availability                                                       |
|                                                                                                |                                                                                |                                                                                     | MOS 5.1.2 Ground Repair Time                                                             |
|                                                                                                |                                                                                |                                                                                     | MOS 5.1.3 Maintainer Rating of System Maintainability                                    |
|                                                                                                |                                                                                |                                                                                     | MOS 5.1.4 Mission System Time Between Unscheduled Maintenance                            |

Last, the “AFOTEC Methodology” model used in the E-X airborne multi-sensor platform was adapted to the second test case, the NGS system model. The elements were again reviewed and modified to fit the model's parameters illustrated in Table 10. The elements associated with the NGS system required only minimal modification, with changes limited to simple edits to change names and parameters to match the model since the E-X airborne multi-sensor platform and the NGS system are relatively similar systems. Table 11 summarizes the various components of the AFOTEC Methodology Model and indicates the number of COIs, MOEs, MOPs, and MOSs associated with each case study.

Table 11. Summary Table of AFOTEC Methodology Model Elements for the Two Case Study Models and the Baseline Case Model

| <b>AFOTEC Methodology Model Elements</b> | <b>CSARL</b> | <b>E-X Platform</b> | <b>NGS System</b> |
|------------------------------------------|--------------|---------------------|-------------------|
| Critical Operational Issue (COI)         | 3            | 5                   | 5                 |
| Measure of Effectiveness (MOE)           | 4            | 12                  | 12                |
| Measure of Performance (MOP)             | 15           | 18                  | 18                |
| Measure of Suitability (MOS)             | 7            | 21                  | 21                |

#### ***6.2.6.1 Methods Summary***

In summary, the two case study models and the one baseline case model, created using MDTD processes to generate model-driven test plans from MBSE-enabled system models, were serially developed and then assessed to evaluate the effectiveness and applicability of the proposed AFOTEC Methodology. The outcome of this process is model-based descriptions of the elements within the “AFOTEC Methodology” model, which are used to develop an operational test plan for any SUT.

### ***6.2.7 Results***

The results first compare the baseline CSARL model to the E-X model, focusing on the MR% and RVA metrics of reuse. Next, the E-X platform model is compared to the NGS system model. Once again, the MR% and RVA metrics are assessed and discussed. Lastly, the RVA metric of reuse is evaluated between the baseline CSARL case model and the second test case NGS system model. These results illustrate the function of the proposed MDTD methodology and indicate the degree of reuse between the models.

#### ***6.2.7.1 Reuse Between Systems with Different Missions (CSARL and E-X Platform)***

Comparing the elements and structures of the baseline CSARL (Table 8) and the E-X platform models (Table 9), we observed that the “AFOTEC Methodology” model folder structure and model architecture, as shown in Figure 20, remained the same. However, adapting the folder structure and model architecture to the E-X platform model required adjustments within the “AFOTEC Methodology” model. To accommodate the additional COIs needed for the capabilities of the E-X platform, we duplicated the existing COI 1 from the baseline CSARL case model to develop new COIs for the E-X platform model. We modified the duplicated COI’s contents to reflect the information needed for the new COI in the E-X platform model. The E-X platform model incorporates modified versions of the original three CSARL COIs and two additional COIs created using this duplication process. Similarly, the existing MOE 1 from the CSARL model was repeatedly duplicated, reconfigured, and modified to align with the E-X platform model’s requirements, creating eight more MOEs. This duplication process was also applied to the MOP and MOS sections of the E-X platform model, yielding three more MOPs and 14 more MOSs for the E-X platform model.



The replication and modification process used to adapt the “AFOTEC Methodology” model illustrates the model’s architectural adaptability to meet the specific demands of the E-X platform model. This process highlighted the importance of reuse in streamlining the development and creation of new structures within a model. Moreover, this approach maintains consistency among individual elements across models, aids in clarity, and reduces ambiguities in information by reusing aspects within the “AFOTEC Methodology” model. However, caution must be taken when reusing models, as applying them carelessly to a new SUT can introduce significant risks if not carefully considered for the new context.

#### ***6.2.7.2 Reuse Between Systems with Similar Missions (E-X Platform and NGS System)***

When the AFOTEC test structure was developed for the E-X platform (Table 9), it was then adapted to the NGS system (Table 10), the folder structure and model architecture again remained consistent. Given the similarity in mission between the two models, many analogous functions and related behaviors are shared in the two models. As such, transferring the “AFOTEC Methodology” model to the new model in the “AFOTEC Methodology” folder necessitated only minor adjustments to subsequent elements in the design folder. The COIs remained unchanged, with only minor terminology adjustments to reflect the NGS system's components. Similarly, adapting the MOEs, MOPs, and MOSs involved renaming and updating elements while significantly reusing model architectures and elements. Since the E-X platform and NGS system models are very similar, no additional MOEs, MOPs, or MOSs were added to the NGS system model and applied to the test plan.

As before, the duplication and modification processes were used to adapt the “AFOTEC Methodology” model from the E-X platform model to the requirements of the NGS system model.

This reuse process for developing and creating new structures within a model was straightforward to implement. As with the previous case, the methodology ensures consistency among individual elements across models, enhances clarity, and minimizes uncertainties by reusing aspects within the “AFOTEC Methodology” model. As stated previously, careful consideration is necessary when reusing models, as improper application to a new SUT can lead to significant risks if not thoughtfully adapted to the new environment.

### ***6.2.8 Discussion***

#### ***6.2.8.1 Implications of Reuse Applied to the Case Studies***

Model reuse is often cited as a primary mechanism by which MBSE processes (including MDTD) can realize efficiencies and improvements [61]. Still, research literature has rarely measured the degree of reuse and its value [62]. A few studies have defined reuse in SysML models in toy problems [61] or product families [63], but none measure reuse outside of the architectural and design stages (as this study does in the testing stage) of the MBSE lifecycle.

When comparing the types of reuse present between the baseline CSARL and first test case E-X platform models, we found that reuse predominantly fell into the Modified Subset Reuse category, specifically within the Major Modifications sub-category, as the process involved extensively modifying and duplicating elements to meet the E-X platform model's requirements. Conversely, when examining the E-X platform and NGS system models, reuse predominantly aligned with the Modified Subset Reuse category but within the Minor Modifications sub-category, where adjustments were limited to name changes required to integrate seamlessly with the NGS system model.

Quantitatively, Table 12 shows the percentage of element reuse between the baseline CSARL and first test case E-X model, with the application associated with each case study

significantly different. The average percentage of element reuse was 25.5%. Table 12 displays the values for the RVA metric, with the average being  $RVA = 1.72$ , indicating that the E-X platform test case model is 72% more effective than if the product was created without reuse [58].

Table 12. Reuse of Model Elements Between the Baseline CSARL and the E-X Platform Case Study Models

| Case Study Comparison            | CSARL vs. E-X Platform Percent Reuse (MR%) | CSARL vs. E-X Platform RVA                  |
|----------------------------------|--------------------------------------------|---------------------------------------------|
| Critical Operational Issue (COI) | $\frac{3}{3 + 5} * 100 = 37.5\%$           | $\frac{(5 + 3) + [1 \times 3]}{5} = 2.2$    |
| Measure of Effectiveness (MOE)   | $\frac{4}{4 + 12} * 100 = 25.0\%$          | $\frac{(12 + 4) + [1 \times 4]}{12} = 1.67$ |
| Measure of Performance (MOP)     | $\frac{3}{3 + 18} * 100 = 14.3\%$          | $\frac{(18 + 3) + [1 \times 3]}{18} = 1.33$ |
| Measure of Suitability (MOS)     | $\frac{7}{7 + 21} * 100 = 25.0\%$          | $\frac{(21 + 7) + [1 \times 7]}{21} = 1.67$ |
| <b>Average</b>                   | 25.5%                                      | 1.72                                        |

Similarly, the data presented in Table 13 highlights the percentage of element reuse between the E-X platform and the NGS system models. Additionally, the table shows the RVA metric between these two models and between the second test case, the NGS system model, and the baseline CSARL model, which is one generation older. The average element reuse percentage between the E-X platform and the NGS system models is 100%, demonstrating the E-X test case model is completely reused. The average RVA metric between the E-X platform and the NGS system models is 2.0, indicating that the model is twice as effective than if the model was created without reuse [58]. Furthermore, the RVA metric between the NGS system model and the baseline CSARL model averages 2.08, indicating that the NGS system model is over twice as effective as if the model was created without reuse [58] despite the generational difference.

Table 13. Reuse of Model Elements Between the E-X Platform and the NGS System Case Study Models

| Case Study Comparison            | E-X Platform vs. NGS System<br>Percent Reuse<br>(MR%) | E-X Platform vs. NGS System<br>(RVA)        | CSARL vs. NGS System<br>(RVA)              |
|----------------------------------|-------------------------------------------------------|---------------------------------------------|--------------------------------------------|
| Critical Operational Issue (COI) | $\frac{5}{0 + 5} * 100 = 100\%$                       | $\frac{(5 + 0) + [1 \times 5]}{5} = 2.0$    | $\frac{(5 + 3) + [2 \times 3]}{5} = 2.8$   |
| Measure of Effectiveness (MOE)   | $\frac{12}{0 + 12} * 100 = 100\%$                     | $\frac{(12 + 0) + [1 \times 12]}{12} = 2.0$ | $\frac{(12 + 4) + [2 \times 4]}{12} = 2.0$ |
| Measure of Performance (MOP)     | $\frac{18}{0 + 18} * 100 = 100\%$                     | $\frac{(18 + 0) + [1 \times 18]}{18} = 2.0$ | $\frac{(18 + 3) + [2 \times 3]}{18} = 1.5$ |
| Measure of Suitability (MOS)     | $\frac{21}{21 + 0} * 100 = 100\%$                     | $\frac{(21 + 0) + [1 \times 21]}{21} = 2.0$ | $\frac{(21 + 7) + [2 \times 7]}{21} = 2.0$ |
| <b>Average</b>                   | 100%                                                  | 2.0                                         | 2.08                                       |

Since the E-X platform and the NGS system models are similar, and the model changes from the E-X platform to the NGS system are primarily limited to name changes, it is no surprise that the reusability of elements from the E-X model to the NGS model is 100%. However, this formula does not distinguish between identical subset and modified subset reuse or account for the two subcategories in the modified subset reuse category. This distinction is crucial, as it helps us understand the nuances of reuse. So, although these results indicate a high degree of reuse, these reuse metrics are perhaps not perfectly characteristic of reuse's ease or overarching value. By distinguishing between the Identical and Modified Subset Reuse categories and further discriminating between minor changes and significant modifications within the Modified Subset Reuse category, the detailed results can illustrate a complete accounting of the degree of reuse available under MDTD constructs.

#### ***6.2.8.2 Implications for the Model-Driven Test Design***

These test models demonstrate that the MDTD development process is effective for the case study's SUTs and can be applied to simple and complex MBSE artifacts. In the authors' assessment, these case study results show that even for very different projects with low apparent similarity, adapting the AFOTEC MDTD Methodology proved more feasible than expected, highlighting the utility of incremental model-driven systems engineering and MDTD.

These findings also have some implications for developing strategies for transitioning to MBSE and Product Line Engineering constructs. This research proves that even simple MBSE projects can lead to an improved understanding of the processes and the building of a well-documented library of models ready for reuse. This study shows that even in very different modeling and application domains, we achieved a 25.5% reuse rate between the testing models, indicating a clear pathway to effective model-driven test development. However, these findings can be limited by a shortage of skilled personnel and organizational resistance to adopting MBSE practices [64]. A significant challenge in realizing the potential of MBSE is the lack of expertise and the lack of necessary tools for effective implementation [64], which hampers the development of well-designed, reusable models.

One critique of measuring reuse with the RVA metric is that it places equal value on reusing lower-importance, less-frequently executed code as it does on high-importance and frequently executed code, which may bias reuse measurements for complicated programs. In using RVA to measure the reuse of model artifacts, we have only measured the reuse within the testing model of high-value components like COIs, MOEs, MOPs, and MOSs, which are essential for developing test methodology models. RVA, therefore, measures the reuse of the model's core elements,

thereby promoting the reuse of the most valuable artifacts. As a result, in our case studies, we have applied RVA metrics selectively and strategically, focusing only on the most essential elements.

This approach provides a foundation for effective model-driven test development, demonstrating the applicability of the AFOTEC MDTD methodology and the potential for successful model reuse. The two case studies and the baseline case presented in the paper indicate that by utilizing the MBSE-enabled MDTD process to build test models, we can achieve a reuse rate of model components between 25.5% and 100%. Using the RVA metric to measure reuse efficiency, we find that models created with reuse can be twice as effective as those developed without reuse. Thus, the reuse of even simple MBSE models plays a crucial role in enhancing the total effectiveness of program development, with evidence from this study showing benefits in MDTD example cases.

#### ***6.2.8.3 Discussion Summary***

In the context of MDTD, this substantial reduction in effort results from the ability to derive and reuse test models and test plans from a model-driven test development process. Reusing these test models and plans significantly reduces the effort required to develop test models for future SUTs. This approach fulfills and aligns with many of the goals of the DoD Digital Transformation process. Despite these advantages, a shortage of skilled personnel and resistance within organizations can limit the adoption of MBSE practices. One of the main challenges to fully leveraging MBSE is the lack of expertise and appropriate tools for successful implementation, which restricts the creation of well-designed, reusable models [64].

### **6.2.9 Conclusion**

These two test cases and the baseline have demonstrated that using the “AFOTEC Methodology” model to generate elements needed for developing and creating test plans can be applied to a set of applications and complex SUTs. The extent to which the reuse of modeling components can be enabled depends on the differences between the models. Still, essential elements can be duplicated and modified to meet the new model’s requirements. Metrics of model reuse can be challenging and may not be a reliable indicator. Therefore, greater granularity must be incorporated into the calculation to achieve a metric that effectively spans a broader range of values. Although MBSE offers clear advantages, a shortage of skilled personnel and organizational resistance can hinder its adoption. A key obstacle to fully realizing the benefits of MBSE is the lack of expertise and necessary tools for effective implementation, which limits the development of robust, reusable models. Implementing the AFOTEC test methodology using a model-based approach ensures that test planning and documentation can take advantage of MBSE artifacts accompanying future flight test systems. This approach enables test plan component reuse, with corresponding benefits in configuration management, consistency, and standardization across the DoD testing enterprise.

### **6.3 Addressing Research Question 2**

After performing the above research tasks, I am ready to address Research Question 2, which is restated as follows: *What are the costs and benefits of the techniques associated with Research Question 1 when implemented in an MBSE construct?*

This review found that two reuse metrics, RM% and RVA, can be used to evaluate and describe reusability across different generational models. Two MBSE case studies and one baseline case study developed within an MDTD framework demonstrated the applicability of these

metrics and were assessed for their costs and benefits regarding the reusability of models and artifacts. The initial model, the Combat Search and Rescue Locator (CSARL) was created to establish a model-based AFOTEC test methodology (Figure 20). The other two models, the E-X airborne multi-sensor platform, described in the Effective Model-Based Systems Engineering textbook [33], and the next-generation sensor (NGS) system, a generalized version of a real-world model, were then developed. Subsequently, the AFOTEC test methodology was applied to these two models and evaluated for its reusability within each model and its effectiveness in generating a model-based test plan artifact.

These two test cases and the baseline test case illustrated that employing the AFOTEC test methodology to generate elements necessary for developing and creating test plans can be applied to various situations or conditions. The level of reusability relies on how much the models vary, yet the core components can be replicated and adjusted to suit the needs of the new model. By adopting the “AFOTEC Test Methodology” through a model-based approach, test planning documentation remains current, reducing the need for extensive rewrites whenever changes occur in the model, thus supporting robust configuration management. Additionally, consistency and standardization are achieved over time as the same process is reused across different models. Other benefits include increased accuracy of information and reduced ambiguities and misinterpretations common in written documents.



## **CHAPTER 7 – COSTS AND BENEFITS OF USING A MODEL-BASED APPROACH IN AN EXAMPLE ACQUISITION PLANNING SCENARIO**

### ***7.1 Introduction***

The chapter addresses Research Question 3, which is restated as follows: *Compare the techniques associated with Research Question 2 with the current approach used in flight testing.*

To answer this question, the chapter begins with Section 7.2, which provides a narrative review of the current AFOTEC flight test approach for evaluating SUTs. Section 7.3 then examines the effectiveness and feasibility of developing, creating, and semi-auto-generating test plans from model-based information using the Velocity Template Language (VTL) to create templates within the Dassault Systemes' Cameo Systems Modeler™ software package.

### ***7.2 AFOTEC Flight Test Approach***

The following discussion was presented to the dissertation committee during the initial assessment and details the work accomplished to answer research question 3, task 1. It focuses on characterizing the current AFOTEC flight-testing approach.

#### ***7.2.1 Test Planning Process***

Proper execution of test and evaluation (T&E) and the accomplishment of OT&E is the primary interest of DOT&E before an acquisition program exists in the PD phase and enters the last stage known as Operations and Support and full-rate production (FRP) is granted, the system reaches full operational capability (FOC), and full-rate production (FRP) and deployment begins. DOT&E accomplishes oversight by approving the adequacy of test plans developed, written, and executed by AFOTEC personnel with assistance from DOT&E office personnel.

Test plans are divided into multiple levels and grouped into mission effects, system capabilities, system functions/properties, and system structure. These categories focus on operational testing, emphasizing mission effects and how they are achieved. These system capabilities are the integration of multiple system functions and properties that are combined to achieve the desired mission effect.

### ***7.2.2 Operational Test & Evaluation Construct***

The OT&E Construct is the framework for conducting an operational test. Assessments of operational effectiveness and suitability are the two elements that make up the OT&E Construct, and the framework defines their relationships. Operational effectiveness measures the system's overall ability to accomplish a mission when used by representative personnel in the expected or planned operational environment while considering organization, doctrine, supportability, survivability, vulnerability, and threat. Operational suitability is the degree to which a system can be placed satisfactorily for use in the field while considering its availability, compatibility, transportability, interoperability, reliability, wartime usage rates, maintainability, safety, human factors, manpower supportability, logistics supportability, natural environmental effects, and impacts, documentation, and training requirements.

These two elements of the operational test are assessed by critical operational issues (COI), which address the vital mission elements or operational objectives to determine the system's overall capability to support mission accomplishment. These COIs are assigned measures designed to convey information about a system's effectiveness, capabilities, functions, properties, or structure. COIs are categorized into measures of effectiveness (MOE), which are further refined into measures of performance (MOP) and measures of suitability (MOS). These objectives assess a system's behavior and capability in an operationally relevant environment while achieving an

objective or creating an effect. A measure of effectiveness (MOE) is designed to assess the degree to which the System under test (SUT) can accomplish mission objectives and achieve desired mission effects. A measure of performance (MOP) is used to quantify a system's capabilities. In contrast, a measure of suitability (MOS) assesses the system's ability to support the SUT in its intended operational environment [45]. Furthermore, a data element is defined as any measurement or piece of information used to determine the value of a measure. Data elements may be quantitative, qualitative, subjective, or objective. Figure 21 illustrates the hierarchy of these measures.

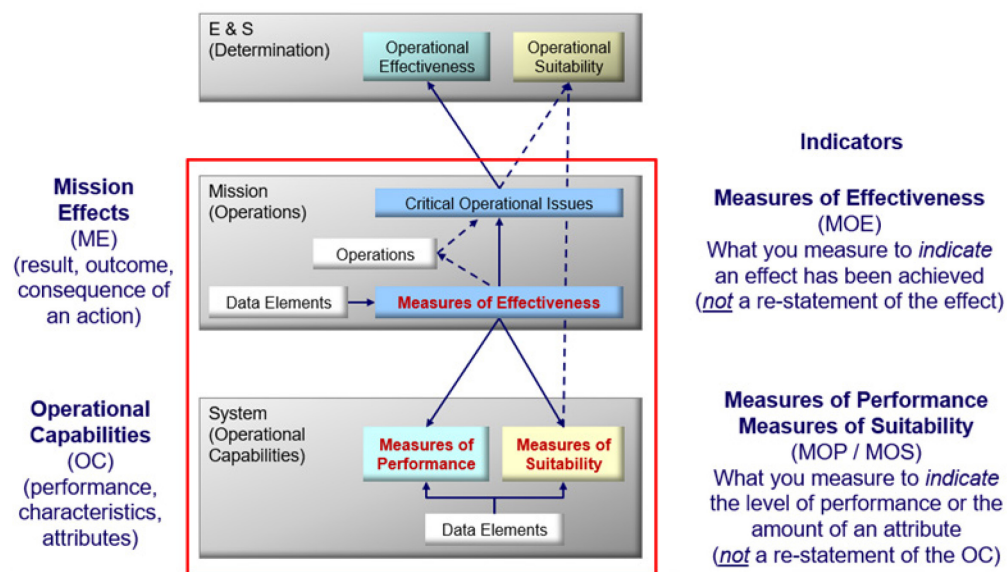


Figure 21. Measure Hierarchies and the Overall AFOTEC OT&E Construct.

### 7.2.3 Test Design

Each COI is assigned at least one Mission Effect, a crucial indicator that a SUT can accomplish its mission in the operational environment. These Mission Effects, the dependent variables that guide the Battlespace Development process, are determined by participants from

various test organizations and subject matter experts from the user community. Battlespace Development aims to take these mission effects and assign factors set at different levels that vary during a test or experiment to determine mission effectiveness or suitability. These Factors are considered the independent variables that define the battlespace with a range of values called levels that may affect a Mission Effect when the factor's level is changed [32].

The last consideration is the magnitude of a factor's impact on each Mission Effect and the likelihood of each factor level occurring in the operational environment. Four levels are considered to determine the effect [32].

- *Vary*, where the factor is deliberately manipulated during the test.
- *Fix*, where the factor level is held constant during the test.
- *Log*, where the level of this factor is recorded as observation during a test run.
- *Demo*, where the level is incorporated into a test run that is executed and analyzed separately from the design, incorporating the varied factors.

Once all COIs, MOEs, MOPs, MOSs, Mission Effects, factors, and levels have been determined, and a test design workbook (TDW) summarizes all the test planning elements. Still, before selecting a test design, the goal of the test must be established to determine how the test is designed and executed and how the information will be reported. Within AFOTEC, the three goals of a test are to characterize system performance, compare system performance to a requirement or another system, and identify problems with the system [32]. A separate test design for each COI is typical, as factors and levels often differ between operations.

The AFOTEC process to execute an operationally sufficient and technically adequate test is grounded in the seven-step design of experiments process, which is [65]:

1. Recognize and state the problem.
2. Select the response variable.
3. Choose the factors, levels, and ranges.
4. Choose the experimental design.
5. Perform the test.
6. Analyze the data.
7. Conclusions and make recommendations.

Test programs use a combination of designs or a separate test design for each operation under a COI, as factors and levels often differ between operations. Some Mission Effects may even require a separate test design. These test designs have associated quality indicators, referred to as measures of merit, which provide a basis for comparing multiple designs. Furthermore, measures of merit are used as evidence that a test design meets the AFOTEC standards for technical adequacy [32]. Most test programs typically input the TDW file information into a statistical software package, where different designs are generated, and their strengths and weaknesses are analyzed for comparison. The most common designs used include full and fractional factorials, central composite, optimal design, and augmented designs, with the primary goal being the characterization of the main effects and most two-way interactions by carefully considering the resolution of the design (especially in factorial designs), to minimize confusion between effects of interest [65].

The test planning process that AFOTEC uses is well-established, as outlined in references [49] and [32]. However, the flight-testing community has not implemented this process in a digital construct nor with acquisition systems developed using an MBSE approach. The knowledge gained during the review of the AFOTEC test planning process (as documented in Chapters 2 and

7) led to the realization that using developed SysML diagrams employed in MBSE, test scenarios and cases can be developed, and the associated variables can be extracted from state model diagrams, which can then be used as inputs to the planning process to create test plans and various testing activities of a SUT. This chapter seeks to develop examples and assessments using model-based abstractions and implementation layers to create sufficient test scenarios and cases from a given model.

### ***7.3 Publication Details***

The subsequent discussion includes content from my third publication, which was also submitted to the International Test and Evaluation Association (ITEA) Journal of Test and Evaluation. The abstract was selected for presentation in the technical poster paper session of the annual International Test and Evaluation Symposium in Huntsville, Alabama, scheduled for November 2024. This work is fully replicated in this dissertation, with modifications made to meet dissertation style guidelines. Additionally, the references within the publication have been incorporated into the reference section of this dissertation. The following discussion contains the effort conducted after the second publication and details the work accomplished to answer research question 3, task 2. The discussion focuses on characterizing the usefulness and suitability of the auto-generated draft test plans that could be further refined and edited for final coordination and approval following the AFOTEC test planning process [32].

#### ***7.3.1 Abstract***

The Department of Defense (DoD) Test and Evaluation (T&E) community is advancing digital engineering by adopting and developing model-based testing methodologies within the flight test community. This article expands upon the existing grey box model-driven test design (MDTD) approaches and incorporates flight test information into Systems Modeling Language

(SysML) models. These models leverage model-based systems engineering (MBSE) artifacts to generate flight test plans. This article introduces a methodology developed by the Air Force Operational Test and Evaluation Center (AFOTEC) to guide the operational test and evaluation (OT&E) test planning process. This methodology uses MBSE to create, develop, and maintain the information required for test plans within a digital model construct. Three case studies demonstrate the methodology using SysML implemented through an MDTD process and showcase the methodology's applicability to different systems under test (SUT). The results include a set of System Usability Survey (SUS) metrics that measure the method's ability to utilize SysML elements within the model and assess its usability and effectiveness in generating flight test plans. The paper also discusses the method's applicability to various scenarios, the benefits of model-based testing, and its relevance in the context of operational flight testing.

### ***7.3.2 Introduction***

Flight test plans are developed to "communicate technical details and logistics for executing and reporting the outcomes of flight, ground, and laboratory tests on air vehicles, subsystems, and components" [16]. They establish a connection between test points and specified requirements, ensuring that the system under test (SUT) aligns with system specifications and delivers the intended performance in the proposed operational environment. These plans cover all aspects of conducting tests, collecting data, and performing post-flight analysis. Test plans capture and communicate a broad set of derived requirements necessary for successfully executing flight tests and are an integral aspect of the DoD acquisition process [3]. The conclusions drawn from the post-flight data analysis are then incorporated into reports to communicate and validate the SUT's flight test performance [45].

Within the Department of Defense (DoD) digital engineering-driven acquisition process, this research seeks to develop a model-based testing (MBT) methodology using MBSE concepts. This methodology enables the integration of flight test planning into models created using the Dassault Systemes Cameo Modeler™ software. This chapter expands on existing model-based grey box testing processes [28] and incorporating flight test information into SysML models (Reference Chapter 6). The results of these processes create a set of flight-test-specific digital engineering models that represent the product of a DoD MBSE-enabled product development process. These models are scrutinized to assess their adequacy in providing the information necessary for test planning. The MBT process identifies missing parts or information by reviewing the native digital template and the resulting draft test plan.

The proposed MBT process seeks to allow for the re-population and re-generation of a new version of the test plan when new information, updates, or technology is inserted into the model. The costs and benefits of this process are assessed by testing the MBT methodology's ability to be updateable, consistent, and adaptable across acquisition programs [28]. The flight test plans created and developed using this process contain the necessary information to conduct a flight test evaluation of the SUT. The flight test plan collects the required information to populate the three significant sections of the test plan: the front matter, system description and details, and the appendix:

- The front matter contains all program information and mission-critical information.
- The system description describes the overall operations of the system and the significant components that make up the SUT.



- The appendix contains all the critical operating issues (COIs), measures of effectiveness (MOEs), measures of performance (MOPs), and measures of stability (MOSs) pertinent to the testing of the SUT.

### ***7.3.3 Background***

The existing flight test planning procedures used by the acquisition process are meticulous and heavily focused on developing documentation, requiring extensive configuration management to keep essential documents up to date with the SUT. Flight testing considerations typically emerge late in the acquisition process, often leading to program delays due to the time needed for test planning, instrumentation, test operations, and analysis. Generally, flight test planning relies on consensus and expertise-driven decision-making, consuming significant man-hours and incurring high costs. Innovative frameworks such as MBSE and its subsets, MBT and MDTD, are designed to improve the state of the art in flight testing, relative to the default document-centric OT&E procedures.

The MBSE methodology comprises a set of interconnected processes, methods, and tools operating within a "model-based" or "model-driven" framework [34]. This approach generates digital models that serve as abstract representations of reality [36], recalling, and relating associated data through computer models. These models' elements can describe various aspects of a physical SUT's desired structure and behavior, business processes, and requirements. In practice, the set of interconnected state machine diagrams, use-case diagrams, sequence diagrams, and mathematical models allow testers to create independent test cases, scenarios, and plans detached from specific design implementations [47] [38].

Under the MBT paradigm, a subset of the emerging MBSE domain, the model can precisely define test parameters and automatically generate detailed test plans for specific test

cases. This approach facilitates testing, investigation, and optimization, reducing the reliance on physical assets. An MBSE testing model enables the representation, communication, and parametric optimization of various test parameters, allowing for reuse without the need to rewrite the entire test plan. Additionally, MBSE improves the management of system complexity by enabling the system model to be examined from multiple viewpoints and analyzing the impact of changes. It also ensures product quality by providing a system model that can be assessed for consistency, correctness, and completeness [17].

A component of MBT is Model-Driven Test Design (MDTD), which seeks to develop a comprehensive understanding of the relationships and dependencies that govern test design, thereby facilitating a thorough assessment of the testing implications of changes made at any stage of the product's design cycle [48]. MDTD is asserted to significantly improve the overall testing development process significantly [48]. The MDTD process allows a subset of experienced testers to handle the high-level, statistical, and defensibility aspects of test design and development. Meanwhile, other testers plan the details of Validation and Verification (V&V) criteria, testing operations, and safety. This abstraction allows for the early involvement of designers responsible for the test planning process so that they can incorporate testing requirements into the system design [49]. Ideally, MDTD empowers the operators and test team members to influence and contribute expertise in shaping and refining different aspects of a physical SUT's intended structure and behavior.

#### ***7.3.4 Methods***

This section will introduce the two use case models used to create and develop the survey test plan and to evaluate the subsequent survey data. Additionally, an overview of the System Usability Survey (SUS) metric applied to evaluate the survey data will be provided.

#### ***7.3.4.1 Case Studies***

Two case studies were developed and produced using SysML implemented in MBSE with the Dassault Systemes' Cameo Systems Modeler™ software package to create, develop, and assess test plans using the AFOTEC Methodology process. The author created the first case study model, the Combat Search and Rescue Locator (CSARL), was created by the author from the system design and development point of view. The second model, the E-X airborne multi-sensor platform, is a revision and update of an example in the textbook "Effective Model-Based Systems Engineering" [35]. Details of the two case studies used are presented in the following paragraphs<sup>2</sup>.

The first case study focuses on AFOTEC's Combat Search and Rescue Locator (CSARL) system. CSARL is a fictional system developed as a training case for AFOTEC flight test personnel learning to create test plans. It is well-documented, well-accepted in the testing community, and has a thorough test plan "answer key." The CSARL system is notionally a portable hand-held Global Positioning System (GPS) receiver system offering enhanced digital moving maps and real-time navigational capabilities for downed aircrew and search and rescue (SAR) forces during combat and peacetime survival scenarios. Its purpose is to complement current GPS-integrated survival radios, enhancing existing survival and rescue navigation capabilities without relying on traditional map-and-compass techniques [41]. CSARL is an example system demonstrating the proposed methods for relevant flight-test test plan generation. Since the system is well-documented and all AFOTEC personnel are familiar with the application and model, it is a good benchmark for applying the grey box-inspired testing technique along with the AFOTEC Methodology process implemented in an MBSE environment to create a flight test plan without direct SME

---

<sup>2</sup> A third case study was created and developed to assess the reusability of these models, but it was not used to evaluate the usability of test plan creation.

intervention. Moreover, it provides a means to compare outputs between the document-centric and model-centric processes.

The second case study uses a system described in the Effective Model-Based Systems Engineering textbook [35]. The textbook introduces the “E-X Airborne Sensor Platform” as a hypothetical system used to exemplify the development of various diagrams for defining and describing software-intensive aerospace systems using System Modeling Language (SysML). The E-X airborne multi-sensor platform gathers intelligence, surveillance, and reconnaissance (ISR) data. Operating in collaboration with ground stations and affiliated organizations, it facilitates the collection and dissemination of ISR products. E-X executes ISR missions and performs aerial imaging and sensing functions [35]. The E-X case study is a valuable example of a moderately complex system, utilizing SysML diagrams readily accessible in the textbook [35] and easily replicated. Like the CSARL case study, the E-X system includes comprehensive documentation, making it an ideal standard for applying grey box-inspired testing techniques and the AFOTEC Methodology process within an MBSE environment. Additionally, it offers another platform for comparing outcomes between document-centric and model-centric processes.

#### ***7.3.4.2 Auto-Generated E-X Airborne Sensor Platform Test Plan***

The E-X case study model was used to generate the Cameo auto-generated (CAG) draft test plan used to conduct the survey and collect the usability data, which was the hypothetical E-X Airborne Sensor Platform system [35]. Its selection for demonstrating flight test-relevant test plan generation methods was intentional, leveraging readily available SysML diagrams that could be easily implemented and reproduced. Once the pertinent diagrams from the E-X system model were identified and reproduced using the Cameo Enterprise Architecture software, the AFOTEC methodology test structure was incorporated and modified as discussed in Chapter 6 to complete

the model for use in the case study. The elements within the structure were assessed for content to ensure that each MOE, MOP, and MOS was linked to their associated COI and were populated with all the essential details to fit the model and meet the AFOTEC test planning guide [45] requirements, as illustrated in Figure 22.

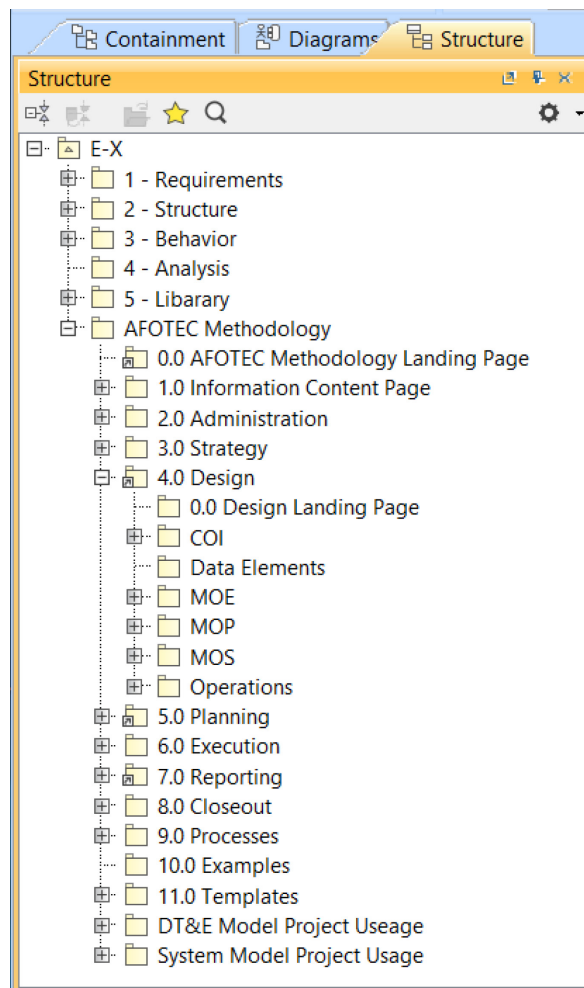


Figure 22. E-X Model Layout Structure Incorporates the AFOTEC Test Model in a Folder Labeled AFOTEC Methodology.

The generation of the test plan documentation process took advantage of the model-generated artifacts Model Development Kit (MDK) contained within the Government Issued Framework (GIF) that uses the weapons government reference architecture (weapons GRA)

developed by The Air Force Life Cycle Management Center. The MDK was modified using the Velocity Template Language (VTL) to create and develop the template to produce the draft test plan.

The author manually tested the modified VTL template from the MDK, which was used on the baseline CSARL model, to verify that the document generated by the template was accurate. To ensure correctness, the author compared the VTL template's output with the original CSARL test plan, confirming that all information was correctly extracted and the output was complete. The modified VTL template was then applied to the E-X test case model to generate a draft test plan.

The E-X model was linked to the MDK as a “project usage” model since the E-X test case model contains all the information required for each COI, MOE, MOP, and MOS associated with the E-X model. The MDK structure was then modified to reflect the sections needed for the AFOTEC test planning guide [45] and other relevant sections. The draft test plan was then developed by running the VTL script, which created the draft test plan document as an output.

#### ***7.3.4.3 System Usability Survey***

The System Usability Survey (SUS), developed in 1986, is a questionnaire designed to assess users' perceptions of a system's usability [66]. The survey uses a set of objective measures to evaluate the impact of specific changes made to a product or service [66]. “The international standard ISO 9241 breaks the measurement of usability down into three separate components that must be defined relative to the context of use” [66]:

- **Effectiveness:** Whether people can complete their tasks and achieve their goals.
- **Efficiency:** The extent to which they expend resources to achieve their goals.
- **Satisfaction:** The level of comfort the experience provides in achieving their goals.

Thus, the objective of the SUS is to provide a measure of a subject's perceptions of the usability of a system or product in a short period during the evaluation [66].

The SUS consists of ten statements, alternating between five that indicate strong agreement and five that indicate strong disagreement, to minimize response bias [66]. The statements are rated on a scale from 1 to 5, with a score of 1 for "strongly disagree" and a score of 5 for "strongly agree" [66]. For each positive statement, the individual score is reduced by 1, and these adjusted scores are then summed. The score is subtracted from 5 for each negative statement, and these adjusted scores are also summed separately. The two sums are added and multiplied by 2.5 to obtain the user's SUS score [66].

#### ***7.3.4.4 Interpretation of the System Usability Survey Score***

The SUS score, developed by Bangor, Kortum, and Miller in 2009, ranges from 0 to 100 [67]. They created a scale, as illustrated in Figure 23, that correlates users' SUS scores with specific adjectives and ratings to provide a grading score, with higher scores indicating better usability. A SUS score of 68 corresponds to the 50th percentile. Scores above 70 are typically considered good, with 85 indicating excellent usability, while scores below 50 suggest poor user experience [67]. Research has shown that SUS can effectively measure a system's perceived usability with a small sample size (around 8-12 users), providing a reliable assessment of how people view your system or product [66]. AFOTEC uses the acceptability ranges and adjective ratings to evaluate a SUT's usability, effectiveness, and suitability. User comments and surveys conducted during the execution of a SUT program are collected to gather survey data.

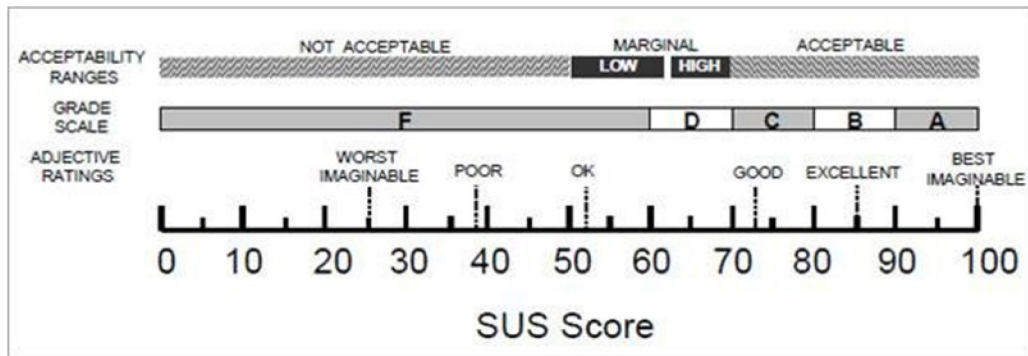


Figure 23. A Comparison of the Adjective Ratings, Acceptability Scores, and School Grading Scales in Relation to the SUS Score [67].

### 7.3.5 Assessment Method

#### 7.3.5.1 Rating the Auto-Generated Draft Test Plan

To evaluate the effectiveness, suitability, and usability of the CAG draft test plan, current AFOTEC members from Detachment 5 were issued a survey. These participants are test and evaluation professionals with varying experience in operations, have different maintenance backgrounds, and were composed of both officer and enlisted personnel. The survey, which included 21 participants, aimed to evaluate the usability, effectiveness, and suitability (measured as value and quality) of test plan development and reuse using the MDTD process. The results were compared to the current AFOTEC guidance. The survey was circulated among AFOTEC members with varying knowledge and experience in creating and reviewing test plans. Participants were asked to rate their expertise in test plan creation and their involvement in reviewing test plans. Demographic data collected during the survey provided insights into the participants' self-assessed levels of knowledge and participation.

First, the survey asked participants to self-judge their proficiency in “coordinating the test planning process” and “reviewing test plans” by selecting from one of the following: beginner, competent, proficient, or expert.” Based on the answers to these two questions, each participant



was characterized by an overall experience level used in the survey analysis results. Table 14 shows the stratification of the determined experience level and the number of participants in each category.

Table 14. Determination of Experience Level Based on Participant's Self-Evaluation.

| <b>Level of Proficiency in</b> |                                               | <b>Overall Experience Level</b> | <b>Number of Participants</b> |
|--------------------------------|-----------------------------------------------|---------------------------------|-------------------------------|
| <b>Reviewing test plans</b>    | <b>Coordinating the test planning process</b> |                                 |                               |
| Beginner                       | Beginner                                      | Beginner                        | 4                             |
| Beginner                       | Competent                                     |                                 |                               |
| Beginner                       | Proficient                                    |                                 |                               |
| Beginner                       | Expert                                        |                                 |                               |
| Competent                      | Beginner                                      |                                 |                               |
| Competent                      | Competent                                     | Competent                       | 5                             |
| Competent                      | Proficient                                    |                                 |                               |
| Competent                      | Expert                                        |                                 |                               |
| Proficient                     | Beginner                                      |                                 |                               |
| Proficient                     | Competent                                     |                                 |                               |
| Proficient                     | Proficient                                    | Proficient                      | 6                             |
| Proficient                     | Expert                                        |                                 |                               |
| Expert                         | Beginner                                      | Expert                          | 6                             |
| Expert                         | Competent                                     |                                 |                               |
| Expert                         | Proficient                                    |                                 |                               |
| Expert                         | Expert                                        |                                 |                               |

The usability assessment section of the survey included the ten standard questions from the SUS, as outlined by Aaron Bangor, Philip T. Kortum, and James T. Miller [67]. Additionally, the survey featured four questions specific to the CAG draft test plan. The first question asked participants to evaluate the value of the plan in developing test plans. In contrast, the remaining questions assessed participants' impressions of the plan's quality by addressing the test plan's content.

To conclude the survey, three final questions were asked of the participants to measure their views on the usability of the CAG draft test plan and their impressions on whether the CAG test plan can reduce the time needed to prepare a final test plan. First, participants were asked if they

believed the CAG draft test plan could shorten the time required to create a test plan. Second, they were asked to estimate the percentage of time savings it would provide. Lastly, participants were asked to give their overall recommendation on using the CAG draft test plan.

Appendix A contains a sample version of the survey. The survey participants were given two weeks to respond to the emailed survey document, and the responses were collated and analyzed. The Institutional Review Board (IRB) at Colorado State University reviewed and approved this research under protocol #5680. It was classified as exempt from full review under the category of exempt human subjects' research.

### **7.3.6 Results**

#### **7.3.6.1 Results of System Usability Survey**

The ten standard questions used in the SUS were analyzed and organized based on the participants' experience levels. The participants' SUS scores were assessed using the adjective rating scale shown in Figure 23. Table 15 illustrates this stratification while Table 16 shows the distribution of the scores by experience level. These data are plotted in Figure 24 using a bar chart to illustrate the distribution and their relationship. These findings suggest that the CAG draft test plan has an overall adjective rating of “Good” or better.

Table 15. Classification of SUS Scores Based on Adjective Rating.

| <b>SUS Score</b> |             | <b>Adjective rating</b> |
|------------------|-------------|-------------------------|
| <b>low</b>       | <b>high</b> |                         |
| 85               | 100         | <b>Best Imaginable</b>  |
| 74               | 84          | <b>Excellent</b>        |
| 54               | 73          | <b>Good</b>             |
| 38               | 53          | <b>Okay</b>             |
| 26               | 37          | <b>Poor</b>             |
| 0                | 25          | <b>Worst Imaginable</b> |

Table 16. Distribution of Participant SUS Scores Based on Adjective Rating and Participant Experience Level.

| Rating           | Beginner | Competent | Proficient | Expert |
|------------------|----------|-----------|------------|--------|
| Best Imaginable  | 0        | 4         | 1          | 3      |
| Excellent        | 1        | 0         | 0          | 0      |
| Good             | 2        | 0         | 2          | 3      |
| Okay             | 1        | 1         | 2          | 0      |
| Poor             | 0        | 0         | 0          | 0      |
| Worst Imaginable | 0        | 0         | 1          | 0      |

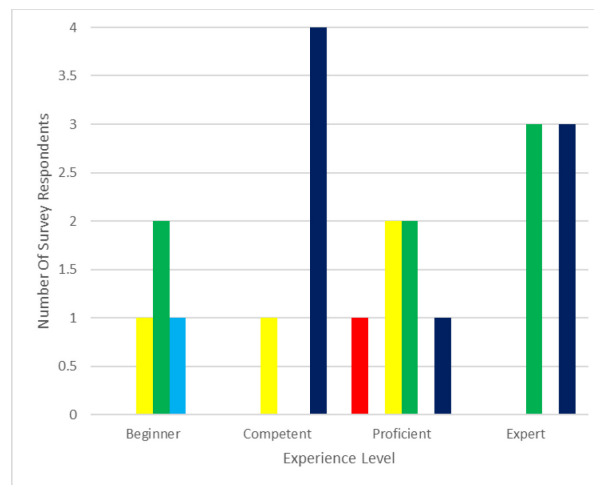


Figure 24. Bar graph Illustrating the Distribution of the Participant's SUS Scores Based Adjective Rating and Experience Level.

The SUS results were evaluated using the acceptability rating scale commonly used by AFOTEC usability of SUTs, as shown in Figure 25. The participants' SUS scores were ranked based on the acceptability rating criteria in Table 17, with Table 18 showing the distribution of scores by experience level. These data are presented in Figure 27 using a bar chart to illustrate the distribution and their relationships. The findings indicate that the acceptability rating for the CAG draft test plan was unclear, with four scores falling in the unacceptable category and four in the low marginal group. However, it's worth noting that the expert experience level category mostly received acceptable ratings.

Table 17. Stratification of SUS Scores Based on Acceptability Rating.

| SUS Score |      | Acceptability rating |
|-----------|------|----------------------|
| low       | high |                      |
| 71        | 100  | Acceptable           |
| 61        | 70   | High Marginal        |
| 51        | 60   | Low Marginal         |
| 0         | 50   | Not Acceptable       |

Table 18. Distribution of Participant SUS Scores Based on Acceptability Rating and Experience Level.

| Rating         | Beginner | Competent | Proficient | Expert |
|----------------|----------|-----------|------------|--------|
| Acceptable     | 1        | 4         | 2          | 5      |
| High Marginal  | 1        | 0         | 0          | 0      |
| Low Marginal   | 1        | 0         | 2          | 1      |
| Not Acceptable | 1        | 1         | 2          | 0      |

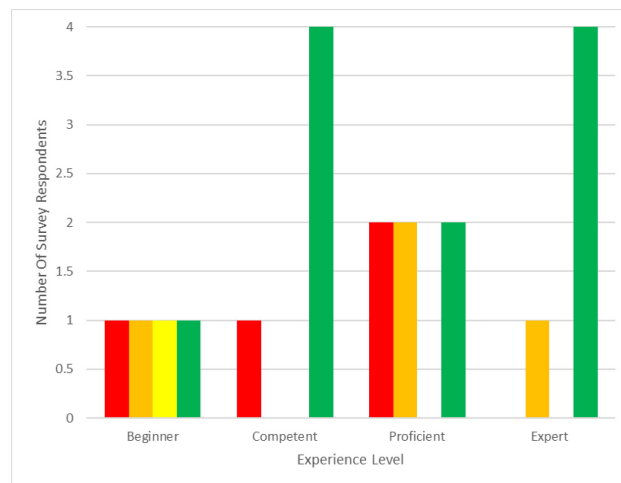


Figure 25. Bar Graph Illustrating the Distribution of the Participant's SUS Scores Based on Acceptability Rating and Experience Level.

The SUS results were evaluated based on the average of all participants' scores, resulting in an average SUS score of 67. Figure 26 demonstrates the distribution of these scores. Since a SUS score of 68 is considered average, a score of 69 suggests that the CAG draft test plan has “slightly above-average” usability.

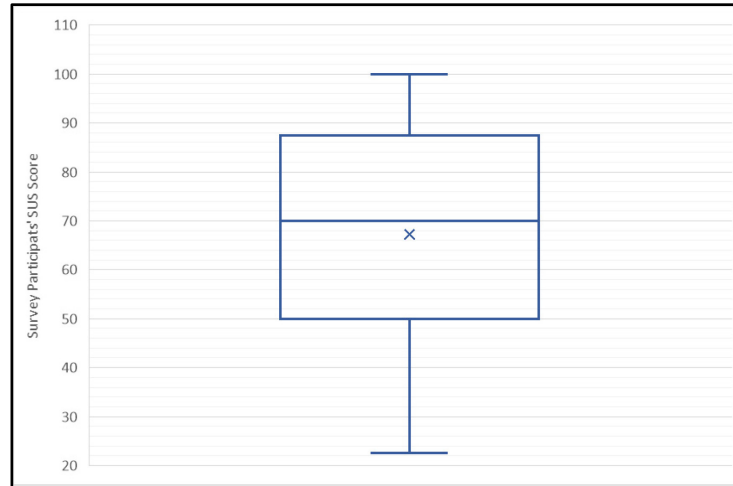


Figure 26. Box Plot Showing the Distribution of the SUS Scores (n=21).

### 7.3.6.2 Analysis of Effectiveness, Value, and Quality

The following discussion presents survey results regarding the CAG draft test plan's effectiveness, value, and quality. The information in the CAG draft test plan is a practical starting point for developing complete test plans that meet AFOTEC's test planning criteria [32].

Table 19. Effectiveness Data Collected During the Survey and Sorted Based on Responses.

| Effectiveness          | Number of Respondents Based on the Following Question                                                   |
|------------------------|---------------------------------------------------------------------------------------------------------|
|                        | “Rate the effectiveness of the CAG test plan as a starting point for developing a completed test plan”. |
| Completely Ineffective | 0                                                                                                       |
| Somewhat Ineffective   | 0                                                                                                       |
| Borderline             | 2                                                                                                       |
| Somewhat Effective     | 10                                                                                                      |
| Completely Effective   | 9                                                                                                       |

Table 19 shows how the survey data is categorized based on participants' opinions about the effectiveness of the CAG test plan as a practical starting point for creating a complete test plan. Of the 21 survey respondents, nine rated the CAG draft test plan as "completely effective." Ten respondents rated it as "somewhat effective," while only two gave a "borderline" rating. Nine of the 21 survey respondents responded that the CAG draft test plan was “completely effective.”

These responses provide evidence that the CAG draft test plan is an “effective” starting point for developing a complete test plan.

Table 20. Value Data Collected During the Survey and Ordered Based on Responses.

| Value                      | Number of Respondents Based on the Following Question |                                                        |                                     |                                                                        |
|----------------------------|-------------------------------------------------------|--------------------------------------------------------|-------------------------------------|------------------------------------------------------------------------|
|                            | “CAG test plan capabilities meet my requirements.”    | “Using the CAG test plan is a frustrating experience.” | “The CAG test plan is easy to use.” | “I must spend too much time correcting things with the CAG test plan”. |
| Strongly Disagree          | 0                                                     | 6                                                      | 1                                   | 3                                                                      |
| Disagree                   | 0                                                     | 4                                                      | 3                                   | 4                                                                      |
| Slightly Disagree          | 3                                                     | 4                                                      | 3                                   | 3                                                                      |
| Neither Agree nor Disagree | 8                                                     | 2                                                      | 3                                   | 5                                                                      |
| Slightly Agree             | 4                                                     | 5                                                      | 5                                   | 6                                                                      |
| Agree                      | 5                                                     | 0                                                      | 6                                   | 0                                                                      |
| Strongly Agree             | 1                                                     | 0                                                      | 0                                   | 0                                                                      |

Table 20 categorizes the survey data based on participants' opinions about the value of the CAG test plan as a tool for developing a complete test plan. The value was assessed through responses to four questions measuring participants' attitudes toward using the CAG draft test plan. The first question asked if the “CAG test plan capabilities meet my requirements.” Of 21 respondents, one “strongly agreed,” five “agreed,” four “slightly agreed,” eight “neither agreed nor disagreed,” and three “slightly disagreed.” The second question asked if “using the CAG test plan is a frustrating experience.” Five “slightly agreed,” two “neither agreed nor disagreed,” four “slightly disagreed,” four “disagreed,” and six “strongly disagreed.” The third question asked if “the CAG test plan is easy to use.” Six “agreed,” five “slightly agreed,” three “neither agreed nor disagreed,” three “slightly disagreed,” three “disagreed,” and one “strongly disagreed.” The final question asked, “I must spend too much time correcting things with the CAG test plan.” Six “slightly agreed,” five “neither agreed nor disagreed,” three “slightly disagreed,” four “disagreed,” and three “strongly disagreed.” These responses provide evidence that the CAG

draft test plan is valuable as a tool for developing a complete test plan. Figure 27 illustrates the distribution of the survey participants' responses to the four questions to assess the CAG draft test plan's value as a tool.

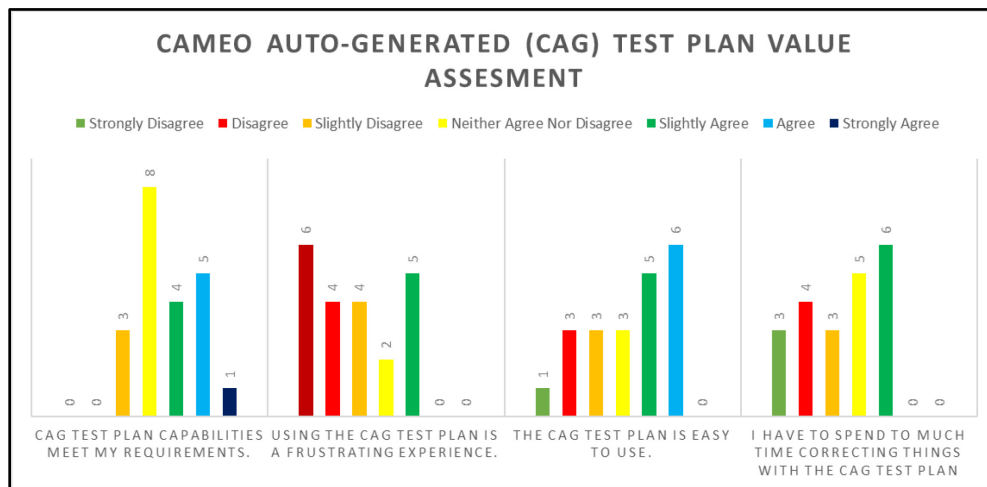


Figure 27. Bar graph Illustrating the Results of the Value of the CAG Draft Test Plan.

Table 21. Quality Data Collected During the Survey and Order Based on Responses.

| Quality                 | Number of Respondents Based on the Following Question |                                 |                    |                          |                                         |
|-------------------------|-------------------------------------------------------|---------------------------------|--------------------|--------------------------|-----------------------------------------|
|                         | Section Headings                                      | Charts, Figures, Diagrams, etc. | Main body contents | Measures section content | Overall acceptability of the CAG format |
| Completely Unacceptable | 1                                                     | 0                               | 0                  | 1                        | 2                                       |
| Slightly Unacceptable   | 2                                                     | 2                               | 2                  | 6                        | 1                                       |
| Borderline              | 3                                                     | 5                               | 3                  | 0                        | 5                                       |
| Somewhat Acceptable     | 7                                                     | 5                               | 10                 | 8                        | 7                                       |
| Completely Acceptable   | 8                                                     | 9                               | 6                  | 6                        | 6                                       |

Table 21 and the bar chart in Figure 28 categorize the survey data based on participants' opinions of the CAG test plan's quality. The quality of the CAG draft test plan was evaluated through feedback on five specific sections. These sections focused on participants' views of the plan's section headings, the presentation of charts, figures, and diagrams, and the contents of the

test plan's main body and measures section. The final feedback area concerned the overall acceptability of the CAG draft test plan's format.

The first area where participants were asked for their opinions was the section headings. Of the 21 participants, eight rated them as "completely acceptable," seven as "somewhat acceptable," three gave a "borderline" response, two found them "slightly unacceptable," and one rated them "completely unacceptable." The second area focused on participants' views of the presentation of charts, figures, and diagrams. Of the 21 participants, nine found this section "completely acceptable," five "somewhat acceptable," five were "borderline," and two found it "somewhat unacceptable." Participants shared their impressions of the main body contents in the third section. Six participants rated it "completely acceptable," ten rated it "somewhat acceptable," three were "borderline," and two found it "slightly unacceptable." The fourth section asked participants about their impressions of the measure section contents. Six rated it as "completely acceptable," eight as "somewhat acceptable," six as "slightly unacceptable," and one as "completely unacceptable." Finally, participants provided their overall impression of the CAG draft test plan's format. Of the 21 participants, six found it "completely acceptable," seven appraised it "somewhat acceptable," five were "borderline," one found it "somewhat unacceptable," and two appraised it "completely unacceptable." These responses indicate the overall quality of the CAG draft test plan is acceptable while highlighting areas where the format meets expectations and where improvements are needed.



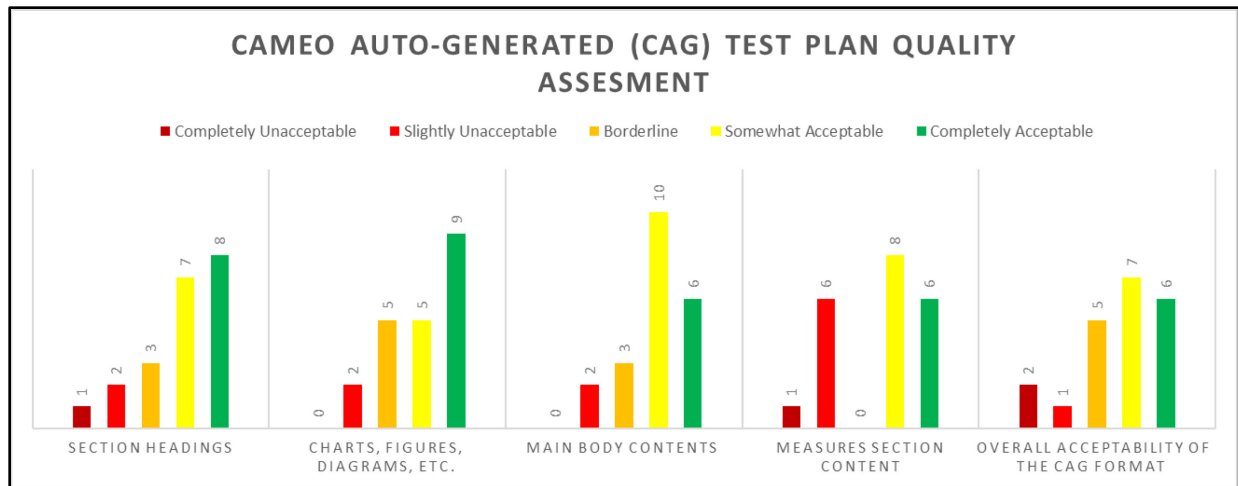


Figure 28. Bar graph Illustrating the Results of the CAG Draft Test Plan Quality.

### 7.3.6.3 Results of Time Reduction and Recommendation of the CAG Draft

#### Test Plan

Table 22 presents the responses to whether using the CAG draft test plan reduces the time required to write a final test plan. All participants, except one, agreed that the CAG draft test plan effectively shortens the time needed to develop a final test plan. When asked to estimate the percentage of time savings the CAG draft test plan could provide, the average response was calculated to be 42%. This metric was further decomposed based on experience level. Table 23 displays the responses inputted by the respondents in terms of the percentage of time savings. This suggests that participants believe the CAG draft test plan can significantly streamline the test plan preparation process.

Table 22. Reduction of Time Achieved by using the CAG Draft Test Plan.

| Reduction of Time | Number of Respondents Based on the Following Question                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------|
|                   | Utilizing the CAG example test plan, do you foresee a reduction of time in writing a final test plan? |
| YES               | 20                                                                                                    |
| NO                | 1                                                                                                     |

Table 23. Reduction Time Achieved by using the CAG Draft Test Plan Decomposed by Experience Level.

| <b>Experience Level</b> | <b>Percent of Average Time Reduction</b> |
|-------------------------|------------------------------------------|
| Beginner                | 46%                                      |
| Competent               | 49%                                      |
| Proficient              | 41%                                      |
| Expert                  | 30%                                      |
| Overall                 | 42%                                      |

Table 24 displays the responses to the question about recommending the CAG draft test plan for use by a test team in writing a final test plan. When asked whether they would recommend the CAG draft test plan as a practical starting point for creating a complete test plan, only two participants did not recommend its use. Once again, the findings suggest that the CAG draft test plan can be a valued tool that can significantly simplify preparing a final test plan.

Table 24. Opinion on Recommending a Test Team for Use in Writing a Final Test Plan.

| <b>Recommendation</b> | <b>Number of Respondents Based on the Following Question</b>       |
|-----------------------|--------------------------------------------------------------------|
|                       | <b>Would you recommend that a test team use the CAG test plan?</b> |
| <b>YES</b>            | 19                                                                 |
| <b>NO</b>             | 2                                                                  |

### ***7.3.7 Discussion***

#### ***7.3.7.1 Benefits of an Auto-Generated Draft Test Plan***

The survey responses from testing professionals indicate that the CAG draft test plan is a practical and recommended tool for creating and developing test plans. Although the average SUS score falls “slightly below average” usability (Figure 29), the survey participants found the CAG test plan effective and valuable. They recommended it for integration into test planning. Moreover, despite the adjective rating of “good” and acceptability rating of “high marginal” (Figure 29), once again, the value and quality of the CAG draft test plan make it a helpful resource, as determined

by the survey participants. AFOTEC members can utilize the CAG draft test plan to prepare suitable test plans for executing tests on a given SUT. The CAG draft test plan is a welcome addition to the test planning process.

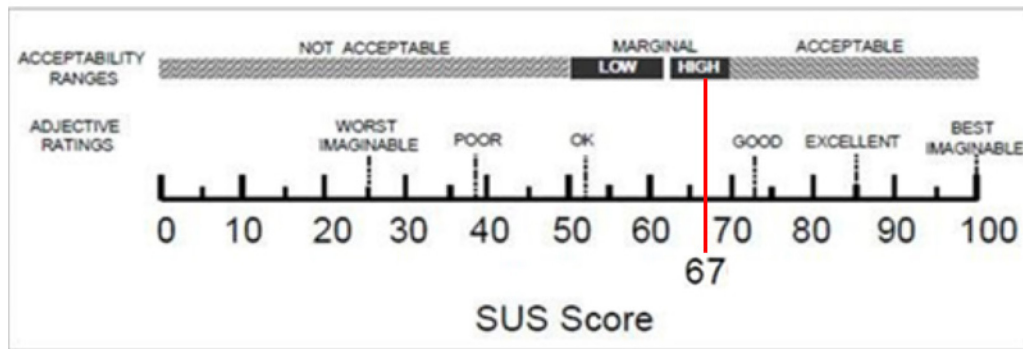


Figure 29. The Red Line Indicates the Calculated Average Participants' SUS Score to the Adjective Ratings, Acceptance Ranges, and the SUS score.

The survey data indicated that the participants believe that the CAG draft test plan can reduce the time it takes a test team to develop a complete test plan by an average of 42%. Although this estimate is based on the opinions of various AFOTEC members, these professionals believe the tool can provide significant time savings. Given that test plans typically take three to nine months to create and develop, and using an average of six months, the average time savings would translate to 75 days saved. According to the survey participants' perception, the CAG test plan results in significant time savings, highlighting the potential efficiency gains it offers. These findings confirm the tool's effectiveness in streamlining the test plan development process.

### ***7.3.7.2 Costs of an Auto-Generated Draft Test Plan***

The final step in the test planning process is to refine the CAG test plan draft through multiple review sessions, during which the test team finalizes the plan to secure stakeholder approval. Once the draft is complete, it undergoes technical editing to ensure compliance with AFOTEC guidelines and formatting standards before receiving final approval and signatures. The more refined the

draft, the quicker this process will be, saving time and resources. Therefore, the closer the auto-generated draft is to a final product, the fewer resources will be required, and the faster the test plan can be approved.

The participants provided valuable feedback through comments and suggestions beyond the survey questions. Although the CAG test plan from this research realized significant potential, the participants identified several issues with the measures section, document formatting, and overall readability. The confusing layout and the unnecessary complexity of modeling MOEs in a BDD add to these challenges. In its current state, the tool may reduce "copy and paste" errors, but the time required to fix output and formatting issues might outweigh the benefits.

To address these concerns, the auto-generated test plan template should be reprogrammed with a stronger focus on formatting and structure to better align with AFOTEC standards. Improvements in readability and layout are also needed. The measures section will require careful attention to organizing and formatting tables for better clarity and usability. Once these issues are resolved, participants recommended additional stakeholder engagement and surveying to assess the CAG's capabilities.

While the auto-generated test plan shows the potential to realize the benefits of MBSE with AFOTEC's test planning approach, improvements will be required to become a fully functional tool. Addressing these issues, especially the confusing layout, and unnecessary complexities, will enhance the tool's efficiency. By refining the template to resolve these problems, the next generations of auto-generated test plans can help streamline the process, reduce the need for extensive revisions, and save time and resources.

### ***7.3.7.3 Costs and Benefits of Implementing a MBSE Approach***

The CAG draft test plan was developed using the MDTD approach within an MBSE framework, proving that MBT combined with MBSE is a viable process for creating and developing test plans. The results in this paper indicate that the CAG draft test plan is beneficial and can be effectively implemented using an MBSE approach. Once a model is created with these processes, AFOTEC members can use the CAG draft test plan to prepare suitable test plans for executing tests on a given SUT. This demonstrates the practical advantages of integrating MBT and MBSE in test plan development. Thus, the CAG draft test plan provides a reliable and efficient method for test planning.

By implementing an MBSE approach, acquisition programs can reduce the time needed to create test plans, significantly lowering the effort required to develop test plans for future systems under test. This approach aligns with the goals of the DoD's digital transformation process. Additionally, it offers tangible benefits in terms of saving effort. Adopting MBSE streamlines test plan development and enhances efficiency, contributing to more effective and timely testing processes.

Implementing the new MBT process within an MBSE framework requires a significant paradigm shift in testers' mindsets. Training personnel is essential to ensure everyone can interpret diagrams effectively. Despite the initial effort, the benefits are substantial. The main advantage is the ability to reuse components, and several secondary benefits follow. Overall, this innovative approach promises enhanced efficiency and effectiveness in test planning.

### ***7.3.8 Conclusion***

The study of an auto-generated draft test plan demonstrates that the model-based AFOTEC MDTD methodology can be applied to complex case studies, such as the E-X model. The study

aimed to create a model that could utilize the AFOTEC MDTD methodology and generate a test plan. The auto-generated test plan was evaluated through a survey of AFOTEC personnel, and the survey participants' perceptions indicated that the auto-generated test plan was beneficial and could be used to prepare suitable test plans for executing tests on a given SUT. This demonstrates the practical advantages of integrating MBT and MBSE in test plan development.

The primary advantage of the current AFOTEC planning process lies in its well-established and accepted nature. However, this process also incurs several costs. These include the requirement to gather and analyze numerous documents to formulate a test plan, the inherent subjectivity in interpreting written materials, and the challenge for testers to determine testing approaches based on incomplete information. Consequently, this often results in frequent revisions, wasting time and resources.

Using the AFOTEC methodology with a model-based approach keeps test planning documentation up-to-date and reduces the time needed to create a test plan. This approach ensures consistency and standardization by applying the same process each time a new test plan is developed. As a result, the time and effort required for test plan creation are minimized. The reuse of the process streamlines documentation and enhances efficiency.

#### ***7.4 Addressing Research Question 3***

After performing the above research tasks, I am ready to address Research Question 3, which is restated as follows: *Compare the techniques associated with Research Question 2 with the current approach used in flight testing.*

This review has shown that semi-auto-generated test plans can be created from model artifacts using templates written in VTL script within the Dassault Systemes' Cameo Systems Modeler™

software package, provided the necessary information to populate the test plan is available within the model artifacts and framework. The CSARL model validated the VTL script that generated the semi-automated draft test plan. Once the script was verified to include all information required by the AFOTEC test planning process [32], it was applied to the E-X airborne multi-sensor platform [35], and the CAG draft test plan was generated. A survey was then conducted using AFOTEC Detachment 5 personnel as respondents, and the results were analyzed to evaluate the usefulness and usability of the CAG draft test plan in accelerating test plan development and improving the overall test planning process.

The survey collected data to assess the effectiveness, format, usefulness, and usability of the CAG draft test plan for AFOTEC test teams. The survey data shows that the AFOTEC methodology, which integrates MBT and MBSE in test plan development, is a viable process for creating and developing test plans and is beneficial in preparing suitable test plans for execution. The CAG draft test plan demonstrated the practical advantages of integrating MBT and MBSE in test plan development. It showed that the draft test planning method provides a reliable and efficient scheme for test planning. Moreover, the survey data indicated that using the draft test planning method can reduce the time a test team takes to develop a complete test plan, highlighting the method's effectiveness in streamlining the development process. This research demonstrates that MBSE can be used to improve the efficiency and effectiveness of test plan development for flight testing.

## CHAPTER 8 – CONCLUSIONS, CONTRIBUTIONS, AND FUTURE WORK

### ***8.1 Introduction***

This research explored software testing techniques and leveraged their processes and metrics for use in flight test T&E. The focus was on employing techniques within the grey box testing domain. The study analyzed two methods, demonstrating that an MBT approach and MDTD processes can facilitate the early establishment of test scenarios and test plan development within a SUT development cycle. The results were comparable to the current document-centric methods. The novel processes discussed in this dissertation offer a foundation for future efforts. They also present a path for AFOTEC to transition from document-centric procedures to model-centric processes, enabling the application of MBSE and its progression into digital engineering.

### ***8.2 Conclusions***

To address Research Question 1: *What software testing techniques are translatable to flight testing?* Chapter 5 delved into the software testing domain and identified that employing techniques within the grey box testing field could facilitate MBT constructs and MDTD methods in flight testing. These methods enabled abstract-level test planning early in the developmental process using SysML-defined activity, sequence, and use case diagrams. A case study demonstrated the potential for efficient and valid use of SysML models in test plan design with MBSE and MDTD [35]. These diagrams were instrumental in determining the number of testing scenarios. The integration of MBSE and grey box testing represents a significant step towards digital transformation and modernizing testing methodologies for improved system evaluation and validation [22] [39].



To address Research Question 2: *What are the costs and benefits of the techniques associated with Research Question 1 when implemented in an MBSE construct?* Chapter 5 provided the foundation to explore task 1 of research question 2, which primarily focuses on defining an MBSE model to realize the methods from research question 1. Chapter 5 detailed the CSARL model used as a test case to develop and test an innovative process for defining and generating testing scenarios. The CSARL case study demonstrates how adopting MBSE methods can aid in formulating test scenarios and can contribute to comprehensive test plans for assessing a system's capabilities [39]. This enhancement can expedite the development of test scenarios [18] [26] by being responsive to mission changes and technology changes that typically occur during a SUT acquisition life cycle.

Chapter 6 further addressed research question 2 by focusing on task 2, which questioned the degree of data reuse for the proposed MBSE plan integration. Chapter 6 discusses the development of an AFOTEC methodology to collect and organize data in a standard AFOTEC test plan. The structure allowed for the semi-automated generation of test plans for editing. The CSARL model was used as a baseline test case to define and create a template for the semi-automated generation of a sample test plan. Two additional test case sample models were then used to evaluate the reusability of the novel process and structure.

The process was applied to a model (E-X platform system [35]) dissimilar in content and function to that of the CSARL. The process was detailed and analyzed, and it was determined that the AFOTEC methodology was reusable due to its modular construction. Artifacts were easily created within a given folder by duplicating and modifying the newly created artifact to meet the needs of the new SUT. The process was then re-applied using the results from the E-X platform model to a second model (NGS System) that was similar in content and function since both models

represented C2ISR systems. Once again, the process was detailed and analyzed, revealing that the AFOTEC methodology remained reusable owing to its modular construction. No new artifacts were deemed necessary, given the similarity between the contents of the two models.

To address Research Question 3: *Compare the techniques associated with Research Question 2 with the current approach used in flight testing.* Chapter 7 detailed the process in which a survey was conducted, and the data analyzed demonstrated that draft test plans can be effectively implemented using an MBSE approach. The survey indicated that once a model is created with these processes, it becomes a viable tool for AFOTEC members to prepare suitable test plans for executing tests on a given SUT. The SUS results indicated that the CAG draft test plan benefits test teams. Furthermore, the data revealed that using these processes to generate draft test plans from models can reduce the time it takes a test team to develop a complete test plan by an average of 42%, providing significant time savings. The process further demonstrates the practical advantages of integrating MBT and MBSE in test plan development, providing a reliable and efficient method for test planning. The following cost and benefits outcomes were determined:

Costs of implementing the process include:

- Implementing the novel process/technique necessitates a change in mindset. Personnel must embrace the concepts and become familiar with the processes and tools used to generate the products.
- Personnel are empowered to learn new processes and tools to implement the novel process/technique. This learning journey will equip them with the skills to read, interpret, and use SysML diagrams effectively to populate the required sections of the AFOTEC methodology structure.

Benefits of implementing the process include:

- The AFOTEC methodology's modular design approach, with its reusability, is a time-saving and efficient solution for creating or modifying products for new, similar, and dissimilar SUTs.
- Implementing this process can reduce the need for multiple SME meetings throughout the project lifecycle. Utilizing SysML diagrams to delineate the behaviors and functions of the SUT can mitigate disagreements among SMEs that may arise from misinterpretations of the system.
- Keeping the process current will significantly reduce the need to conduct multiple reviews of final products, thereby alleviating review fatigue caused by frequent updates to the SUT and enhancing productivity.
- Embracing the paradigm shift toward MBSE in flight test planning aligns with the successful implementation of DoD's Defense Digital Engineering Strategy [30] [33]. Creating a digital test model can incorporate a digital test artifact into a larger life-cycle model [35]. This captures the importance and value of flight and operational testing.

### ***8.3 Research Contributions***

This dissertation has made significant contributions to the fields of MBSE and T&E. Within the MBSE realm, this research focused on the reusability of model elements and the development of metrics to quantify reusability between models. These concepts were demonstrated through two test cases and one baseline case. Within the T&E domain, this research transformed the AFOTEC test planning process into an MBSE construct for implementation in the digital domain. Using the "AFOTEC Methodology" model, an MDTD process created an environment where flight test planning can use SUT models to develop, build, and auto-generate test plans. Survey participants assessed an auto-generated draft test plan, and the results indicated

that using a model-based test planning process tool is a welcome addition to the test planning process and is expected to improve efficiency by providing a practical starting point for creating a final test plan in a shorter time frame.

This dissertation introduces a new methodology for enabling model-based flight test planning processes, significantly contributing to systems engineering and T&E flight testing. A crucial aspect of the research was developing a model-based approach using MDTD grey box testing techniques, which led to the creation of the "AFOTEC Methodology" model. Using a systematic and objective research approach, this model was applied to a simple baseline case study and two relatively complex case studies. As a result, the research generated new insights into reusability and the advancement in the creation of model-based test planning.

Through a literature review, assessment, and case study implementation, the "AFOTEC Methodology" adapted software testing tools to meet the needs of model-based flight testing. This adaptation identified critical testing methodologies successfully demonstrated in a flight test application. Three case studies were developed to create test models that captured the behaviors and functions of SUTs, illustrating how SysML can define and communicate these elements. Additionally, these models generated artifacts that support the creation of flight test plans, improving the efficiency, accuracy, and communication of flight test planning.

This dissertation contributes to the practical assessment of MBSE in flight testing, aligning with the DoD's digital engineering directive. It analyzes the requirements for flight testing hardware and software, uncovering a significant gap between existing software testing capabilities and the specific needs of the Department of the Air Force and aviation test communities. A novel process was developed to reconfigure AFOTEC's workflows using an MBSE framework, improving reusability and streamlining test plan integration. The method enhances flight test

planning using activity, sequence, and use case SysML diagrams while supporting acquisition development within the digital domain.

#### ***8.4 Future Work***

The methodology proposed in this research has significant potential for enhancing the quality of flight test planning and scenario generation using MBT within an MBSE framework for SUT models. This research has demonstrated how these processes can be incorporated into an SUT model and adapted for use. It established the basic methodology, was evaluated through three case studies, and produced outputs like the current document-centric process. However, much work must be done to advance this research further. Future research could extend the concepts in this dissertation by exploring how to incorporate other current AFOTEC processes used in test development, execution, and reporting and transform them into a model-based digital approach. AFOTEC is currently studying integrating the Data Management and Analysis Plan (DMAP) process into the models used in this dissertation and the process for creating the COIs, MOEs, MOPs, and MOSs. The goal is to transform measures and the DMAP processes into a model-based digital approach that can be integrated into any future SUT that AFOTEC evaluates.

Future research could expand the concepts in this dissertation to include DOE and the MBT process to assess test point generation using MOEs, MOPs, and MOSs. This approach can generate data-mining scripts to search test data databases for relevant conditions, enhancing analysis and increasing the statistical power of each test. Incorporating DOE into the MBT process could optimize various parameters within the SUT, resulting in tighter confidence intervals suitable for system verification and validation. Furthermore, exploring these different configurations of artifacts improves knowledge of the decision space and highlights potential excursions.

Another research area that should be explored is combining the DOE and test plan transformation into scripts as input into model and simulation (M&S) testing. This area can be used to examine the area of Digital Twins that they use to help predict system performance and influence system design. By performing focused testing within defined constraints and expectations, systems can be evaluated in a virtual environment that is difficult to replicate or cannot be realized. This virtual testing of a digital representation of the physical SUT bridges the physical and digital worlds. By leveraging these digital models, a means to detect problems or issues can be realized, and insight can be provided to help answer whether the SUT can deliver the operationally intended performance it was designed for. This approach can provide valuable insights into system behavior and performance before the physical system is built or tested.

## REFERENCES

- [1] "IEEE Guide for Software Verification and Validation Plans," in IEEE Std 1059-1993, vol., no., pp.1-87, 28 April 1994, doi: 10.1109/IEEESTD.1994.121430.
- [2] W. L. Jacques and A. S. Gansler, "Using Spiral Development to Reduce Acquisition Cycle Times," 2008.
- [3] Air Education and Training Command, "Air Force Instruction 99-103 - Capabilities-Based Test and Evaluation," 2020.
- [4] "Air Force Digital Transformation," [Online]. Available: <https://usaf.dps.mil/teams/afmcde/SitePages/Functional-Speciality---Test-and-Evaluation.aspx>. [Accessed 28 5 2022].
- [5] "Digital Twins." milWiki. [https://www.milsuite.mil/wiki/Digital\\_Twins](https://www.milsuite.mil/wiki/Digital_Twins) (accessed May 28, 2022).
- [6] E. A. Bjorkman, "Test and Evaluation (T&E): The Authoritative Source of Truth for Digital Engineering," The T&E Exchange, no. 5, p. 3, 2021.
- [7] "Acquisition Process Overview." AcqNotes LLC. [https://acqnotes.com/acqnote/acquisitions/acquisition-process-overview#:~:text=The%20Department%20of%20Defense%20\(DoD,of%20the%20Adaptive%20Acquisition%20Framework%E2%80%9D](https://acqnotes.com/acqnote/acquisitions/acquisition-process-overview#:~:text=The%20Department%20of%20Defense%20(DoD,of%20the%20Adaptive%20Acquisition%20Framework%E2%80%9D). (accessed May 28, 2022).
- [8] "AcqNotes Program Management Tool for Aerospace," AcqNotes LLC, 19 12 2021. [Online]. Available: <https://acqnotes.com/acqnote/acquisitions/emd-phase>. [Accessed 28 5 2022].
- [9] "AcqNotes Program Management Tool for Aerospace." AcqNotes LLC. <https://acqnotes.com/acqnote/careerfields/development-test-and-evaluation> (accessed 11/13/2022).
- [10] "Production & Deployment (PD) Phase." AcqNotes LLC. <https://acqnotes.com/acqnote/acquisitions/production-and-deployment> (accessed May 28, 2022).
- [11] "Operational Test & Evaluation (OT&E)." AcqNotes LLC. <https://acqnotes.com/acqnote/careerfields/operational-test-and-evaluation-ote> (accessed May 28, 2022).
- [12] "Air Force Operational Test and Evaluation Center." Academic. <https://en-academic.com/dic.nsf/enwiki/4569103> (accessed May 28, 2022).
- [13] "Evolutionary Acquisitions." AcqNotes LLC. <https://acqnotes.com/acqnote/acquisitions/evolutionary-acquisitions>, (accessed May 28, 2022).

- [14] W. Roper, "There is no spoon: The new digital acquisition reality," *Defense AR Journal*, vol. 28, no. 4, pp. 488-488, 2021.
- [15] Office of the Deputy Assistant Secretary of Defense for Systems Engineering. *Digital Engineering Strategy*. Department of Defense, Washington D.C., 2018.
- [16] G. Van Peteghem, C. J. Liebmann, S. S. Mailen, J. D. Martin, and K. L. Peck, 2021. *Test Plan Author's Guide*. Engineering Directorate, 412th Test Wing, Edwards AFB.
- [17] INCOSE, 2015. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*. New Jersey: Wiley.
- [18] S. R. Jan, S. T. U. Shah, Z. U. Johar, Y. Shah, and F. Khan, "An innovative approach to investigate various software testing techniques and strategies," *International Journal of Scientific Research in Science, Engineering and Technology (IJSRSET)*, Print ISSN, vol. 23951990, 2016.
- [19] M. A. Jamil, M. Arif, N. S. A. Abubakar, and A. Ahmad, "Software testing techniques: A literature review," in *2016 6th International Conference on Information and Communication Technology for the Muslim World (ICT4M)*, 2016: IEEE, pp. 177-182.
- [20] S. Yoo and M. Harman, "Regression testing minimization, selection, and prioritization: a survey," *Wiley Online Library*, p. 54, 2010.
- [21] N. Anwar and S. Kar, "Review paper on various software testing techniques & strategies," *Global Journal of Computer Science and Technology*, 2019.
- [22] P. Ammann and J. Offutt, *Introduction to software testing*. Cambridge University Press, 2016.
- [23] M. Mussa, S. Ouchani, W. Al Sammane, and A. Hamou-Lhadj, "A survey of model-driven testing techniques," in *2009 Ninth International Conference on Quality Software*, 2009: IEEE, pp. 167-172.
- [24] S. Beydeda, M. Book, and V. Gruhn, *Model-driven software development*. Springer, 2005.
- [25] L. Luo, "Software testing techniques," *Institute for Software Research International Carnegie Mellon University Pittsburgh, PA*, vol. 15232, no. 1-19, p. 19, 2001.
- [26] A. Bansal, "A Comparative study of software testing techniques," *International Journal of Computer Science and Mobile Computing*, vol. 36, no. 6, pp. 579-584, 2014.
- [27] A. A. Sawant, P. H. Bari, and P. Chawan, "Software testing techniques and strategies," *International Journal of Engineering Research and Applications (IJERA)*, vol. 2, no. 3, pp. 980-986, 2012.
- [28] J. L. Alvarado Jr and T. H. Bradley, "Developing Model-Based Flight Test Scenarios," *The ITEA Journal of Test and Evaluation*, vol. 44, no. 4, 2023.



- [29] G. Axiotis, "A new alpha-omega map for acquisition test and evaluation," *International Test and Evaluation Association (ITEA) Journal*, vol.30, no. 4, pp. 473-480, 2009.
- [30] M. C. Collins and M. K. Senechal, "Test and Evaluation as a Continuum," ed: OUSD (R&E), 2023.
- [31] E. Greer, 2010. The integrated T&E continuum is the key to acquisition success. *The Journal of Test and Evaluation*, vol. 31, no. 4, pp. 443-444.
- [32] AFOTEC A-2/9, "Air Force Operational Test and Evaluation Center Test Design Guide,". Albuquerque: AFOTEC, 2018.
- [33] K. Henderson, T. McDermott, and A. Salado, "MBSE adoption experiences in organizations: Lessons learned," *Systems Engineering*, 2023.
- [34] K. Henderson, T. McDermott, E. Van Aken, and A. Salado, "Towards Developing Metrics to Evaluate Digital Engineering," *Systems Engineering*, vol. 26, no. 1, pp. 3-31, 2023.
- [35] J. M. Borky and T. H. Bradley, *Effective Model-Based Systems Engineering*. Springer International Publishing, 2018.
- [36] S. Friedenthal, A. Moore, and R. Steiner, "A practical guide to SysML: the systems modeling language," Morgan Kaufmann, 2014.
- [37] J. Gregory, L. Berthoud, T. Tryfonas, A. Rossignol, and L. Faure, "The long and winding road: MBSE adoption for functional avionics of spacecraft," *Journal of Systems and Software*, vol. 160, p. 110453, 2020.
- [38] Y. Wang and M. Zheng, "Test case generation from uml models," *45th annual midwest instruction and computing symposium, cedar falls, Iowa*, 2012, vol. 4.
- [39] Y. D. Salman, N. L. Hashim, M. M. Rejab, R. Romli, and H. Mohd, "Coverage criteria for test case generation using UML state chart diagram," in *AIP Conference Proceedings*, 2017, vol. 1891, no. 1: AIP Publishing.
- [40] J. Heumann, "Generating test cases from use cases," *The rational edge*, vol. 6, no. 01, 2001.
- [41] Director of Operational Capability Requirements, 2018. Capability Development Document (CDD) for Combat Search and Rescue Locator (CSARL). Washington D.C.: Department of Defense.
- [42] P. B. Nirpal and K. Kale, "A brief overview of software testing metrics," *International Journal on Computer Science and Engineering*, vol. 3, no. 1, pp. 204-211, 2011.
- [43] A. Kaur, B. Suri, and A. Sharma, "Software testing product metrics-A Survey," in *National Conference on Challenges & Opportunities in Information Technology*, 2007, pp. 1-6.

- [44] Department of The United States Air Force, "USAF Deficiency Reporting, Investigation, and Resolution," Secretary of the Air Force, Tinker AFB, OK, 2015.
- [45] AFOTEC A-2/9. Air Force Operational Test and Evaluation Center Test Design Guide. AFOTEC, Albuquerque, 20 July 2018.
- [46] D. Verma, A. M. Madni, S. Hoffenson and L. Xiao, The Proceedings of the 2023 Conference on Systems Engineering Research: Systems Engineering towards a Smart and Sustainable World, Springer, 2024.
- [47] J. Gregory, L. Berthoud, T. Tryfonas, A. Rossignol, and L. Faure, "The long and winding road: MBSE adoption for functional avionics of spacecraft," *Journal of Systems and Software*, vol. 160, p. 110453, 2020.
- [48] M. Mussa, S. Ouchani, W. Al Sammane, and A. Hamou-Lhadj, "A survey of model-driven testing techniques," in 2009 Ninth International Conference on Quality Software, 2009: IEEE, pp. 167-172.
- [49] S. Beydeda, M. Book, and V. Gruhn, *Model-driven software development*. Springer, 2005.
- [50] L. Freeman, P. Beling, K. Esser, P. Wach, G. Kerr, A. Salado, J. Werner and S. Hobson, "Positioning Test and Evaluation for the Digital Paradigm," *The ITEA Journal of Test and Evaluation*, vol. 44, no. 2, June 2023.
- [51] AFOTEC A-2/9, "Air Force Operational Test and Evaluation Center Measures Guide," Albuquerque, 2021.
- [52] "AcqNotes Program Management Tool for Aerospace," AcqNotes LLC, 7 6 2021. [Online]. [Accessed 28 5 2022].
- [53] Q. Wu, D. Gouyon, and E. Levrat, "Maturity assessment of systems engineering reusable assets to facilitate MBSE adoption," *IFAC-PapersOnLine*, vol. 54, no. 1, pp. 851-856, 2021.
- [54] Q. Wu, D. Gouyon, É. Levrat, and S. Boudau, "Use of patterns for know-how reuse in a model-based systems engineering framework," *IEEE Systems Journal*, vol. 14, no. 4, pp. 4765-4776, 2020.
- [55] M. Shinozaki, F. Mhenni, J.-Y. Choley, and A. Ming, "Reuse of SysML model to support innovation in mechatronic systems design," in *2017 Annual IEEE International Systems Conference (SysCon)*, 2017: IEEE, pp. 1-6.
- [56] F. H. Ferreira, E. Y. Nakagawa and R. P. dos Santos, "Reliability in Software-intensive Systems: Challenges, Solutions, and Future Perspectives," in 2021 47th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2021.
- [57] W. Frakes and C. Terry, "Software reuse: metrics and models," *ACM Computing Surveys (CSUR)*, vol. 28, no. 2, pp. 415-435, 1996.

- [58] J. S. Poulin, J. M. Caruso, and D. R. Hancock, "The business case for software reuse," *IBM Systems Journal*, vol. 32, no. 4, pp. 567-594, 1993.
- [59] M. E. Khan and F. Khan, "A comparative study of white box, black box and grey box testing techniques," *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 6, 2012.
- [60] Air Combat Command A5I, "Architecture Data Standards for Next Generation Sensor (NGX)," Department of the Air Force, Washington D.C., 30 Nov 2022.
- [61] F. Wilking, D. Horber, S. Goetz and S. Wartzack, "Utilization of system models in model-based systems engineering: definition, classes and research directions based on a systematic literature review," January 2024. [Online]. Available: <https://www.cambridge.org/core/journals/design-science/article/utilization-of-system-models-in-modelbased-systems-engineering-definition-classes-and-research-directions-based-on-a-systematic-literature-review/04066CE1D254458E17969AC5F2D80347>. [Accessed 18 August 2024].
- [62] A. Albers, N. Bursac and E. Wintergerst, "Product generation development - importance and challenges from a design research perspective.," Proceedings of the International Conference on Mechanical Engineering, pp. 16-21, 2015.
- [63] H. Mause and J. Hummell, "Model-based Product Line Engineering–Enabling Product Families with Variants," IEEE Aerospace Conference, vol. 22, no. 2, pp. 1-8, 2015.
- [64] A. Akundi and W. Ankobiah, "Mapping industry workforce needs to academic curricula - A workforce development effort in model-based systems engineering," Systems Engineering, 2024.
- [65] D. C. Montgomery, Design and Analysis of Experiments. Hoboken: Wiley, 2020.
- [66] J. Brooke, "SUS: a retrospective," Journal of Usability Studies, vol. 8, no. 2, 2013.
- [67] A. Bangor, P. Kortum, and J. Miller, "Determining what individual SUS scores mean: Adding an adjective rating scale," Journal of Usability Studies, vol. 4, no. 3, pp. 114-123, 2009.

## APPENDIX A – USABILITY SURVEY

**Background:** Please note that the draft test plan that I provided, called the “**Cameo Auto-Generated**” (CAG) test plan, is an output of a template that I created using VTL scripting, which I know just enough to be dangerous and create things! As you will see, there are formatting issues, duplicate entries, etc. Most of these issues would be resolved if someone with better programming knowledge of VTL took the template and fixed the errors. Please remember that this is not a “**polished**” **finished product**, and there are misspellings, parts, and pieces missing by design to spark questions and generate suggestions on how to use this “product.”

The vision is for someone (with actual VTL programming experience) to take our tech editors' test plan templates, embed the VTL code that extracts the required information from the test case model, and place that information into a new document in the correct format and order in which we are accustomed to seeing it in our test plans.

Thus, the process will save us time collecting and organizing all the information we put into our test plans and help generate a draft that we can scrutinize for final edits and content during “murder boards.”

**Intent:** The survey I am conducting measures the usefulness of a given test case model's output in generating a test plan. So, the fundamental question I am trying to assess and gather data on is, “Is this ‘product’ useful for AFOTEC to use as a stepping-off point in creating test plans?”

**What I am asking:** Thus, I would like you to consider whether the “CAG” draft test plan has the look, feel, and components required by our test plan process for creating test plans. Can the provided test plan be used as a starting point to begin the test planning process (remember, this is a prototype)?

Your responses to this questionnaire are critical in assessing the Cameo auto-generated test plan. The test plan, “E-X airborne multi-sensor platform Initial Operational Test and Evaluation (IOT&E),” is a fictitious plan developed by Jose Alvarado to demonstrate the capabilities of the SysML software. The information in the E-X test case model was used to provide an example of Dassault Systemes’ Cameo Systems Modeler™ software export product function.

The “auto-generated” test plan is a raw output of the information contained within the model and was extracted using velocity template language (VTL) software and a basic Word document template to develop and create the draft test plan document.

Please provide comments, as they assist in assessing the Cameo auto-generated test plan.

The data results are used for academia for Jose Alvarado’s Colorado State University dissertation “Enhancing Flight Testing Leveraging Software Testing Techniques Implemented in Model-Based Systems Engineering (MBSE).

Your participation in this questionnaire is entirely **voluntary** and will take approximately 30 minutes to complete.

Once the dissertation is completed, a copy can be obtained through a written request from Jose Alvarado.

### **Demographics:**

Full Name: \_\_\_\_\_  
(Last Name) (First Name)

Job/Position Title: \_\_\_\_\_  
(e.g., Lead Engineer, Test Director, Lead Suitability)

Years/months of overall experience in T&E (includes DT & OT): \_\_\_\_\_ / \_\_\_\_\_  
(Years) (Months)

Phone Number: \_\_\_\_\_

Education level: \_\_\_\_\_

### **Questionnaire:**

1. Using the scale below, how would you rate yourself in the following categories:

|                                        | <b>Beginner</b> | <b>Competent</b> | <b>Proficient</b> | <b>Expert</b> |
|----------------------------------------|-----------------|------------------|-------------------|---------------|
| Reviewing test plans                   | <b>0</b>        | <b>0</b>         | <b>0</b>          | <b>0</b>      |
| Coordinating the test planning process | <b>0</b>        | <b>0</b>         | <b>0</b>          | <b>0</b>      |

2. Rate the usability of the “Cameo Auto-Generated “ (CAG) test plan.

| Usability                                                            | Strongly Disagree <span style="float: right;">Strongly Agree</span> |                          |                          |                          |                          |                          |                          |
|----------------------------------------------------------------------|---------------------------------------------------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|                                                                      | 1                                                                   | 2                        | 3                        | 4                        | 5                        | 6                        | 7                        |
| CAG test plan capabilities meet my requirements.                     | <input type="checkbox"/>                                            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Using the CAG test plan is a frustrating experience.                 | <input type="checkbox"/>                                            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| The CAG test plan is easy to use.                                    | <input type="checkbox"/>                                            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| I must spend too much time correcting things with the CAG test plan. | <input type="checkbox"/>                                            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |

3. Rate the acceptability of the “CAG” test plan format.

| Format                                  | Completely Unacceptable  | slightly Unacceptable    | Borderline               | Somewhat Acceptable      | Completely Acceptable    |
|-----------------------------------------|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
| Overall acceptability of the CAG format | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Section headings                        | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Charts, Figures, Diagrams, etc.         | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Main body contents                      | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| Measures section content                | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |

4. Rate the acceptability of the “CAG” test plan content.

| <b>Content</b>                                                             | <b>Completely Unacceptable</b> | <b>Somewhat Unacceptable</b> | <b>Borderline</b>        | <b>Somewhat Acceptable</b> | <b>Completely Acceptable</b> |
|----------------------------------------------------------------------------|--------------------------------|------------------------------|--------------------------|----------------------------|------------------------------|
| Format of the CAG test plan                                                | <input type="checkbox"/>       | <input type="checkbox"/>     | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/>     |
| Consistency of the CAG test plan                                           | <input type="checkbox"/>       | <input type="checkbox"/>     | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/>     |
| Clarity of the CAG test plan                                               | <input type="checkbox"/>       | <input type="checkbox"/>     | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/>     |
| Rate the acceptability of the CAG to follow the AFOTEC Analyst Test Guide. | <input type="checkbox"/>       | <input type="checkbox"/>     | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/>     |

5. Rate the effectiveness of the “CAG” test plan effort.

| <b>Effectiveness</b>                                                                        | <b>Completely Ineffective</b> | <b>Somewhat Ineffective</b> | <b>Borderline</b>        | <b>Somewhat Effective</b> | <b>Completely Effective</b> |
|---------------------------------------------------------------------------------------------|-------------------------------|-----------------------------|--------------------------|---------------------------|-----------------------------|
| Rate the effectiveness of the CAG as a starting point for developing a completed test plan. | <input type="checkbox"/>      | <input type="checkbox"/>    | <input type="checkbox"/> | <input type="checkbox"/>  | <input type="checkbox"/>    |

6. Utilizing the “CAG” example test plan, do you foresee a time reduction in writing a final test plan?

|             | <b>Yes</b>               | <b>No</b>                |
|-------------|--------------------------|--------------------------|
| <b>Time</b> | <input type="checkbox"/> | <input type="checkbox"/> |

7. What percent would a CAG test plan save the test team in preparing a test plan?  
(0 to 100%): \_\_\_\_\_

8. With regards to the “product” (CAG test plan), rate the following items:

|                                                                                             | Strongly Disagree        | Somewhat Disagree        | Neither Agree Nor Disagree | Somewhat Agree           | Strongly Agree           |
|---------------------------------------------------------------------------------------------|--------------------------|--------------------------|----------------------------|--------------------------|--------------------------|
| I think that I would like to use this product frequently.                                   | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I found the product unnecessarily complex.                                                  | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I thought the product was easy to use                                                       | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I think that I would need the support of a technical person to be able to use this product. | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I found that the various functions in this product were well-integrated                     | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I thought that there was too much inconsistency in this product.                            | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I would imagine that most people would learn to use this product very quickly.              | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I found the product very awkward to use                                                     | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I felt very confident using the product.                                                    | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |
| I needed to learn a lot of things before I could get going with this product.               | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/>   | <input type="checkbox"/> | <input type="checkbox"/> |

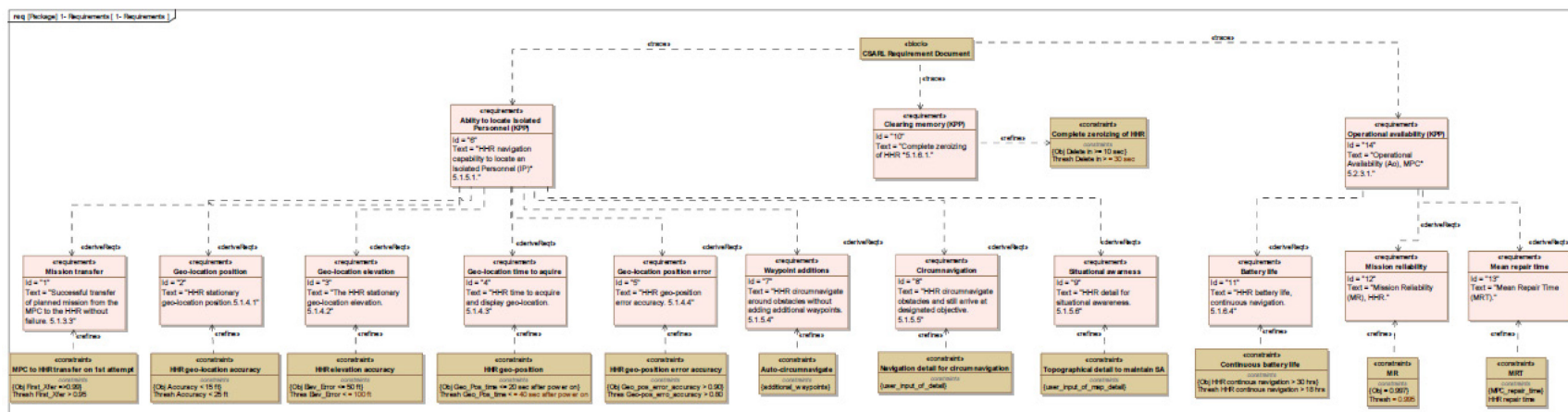


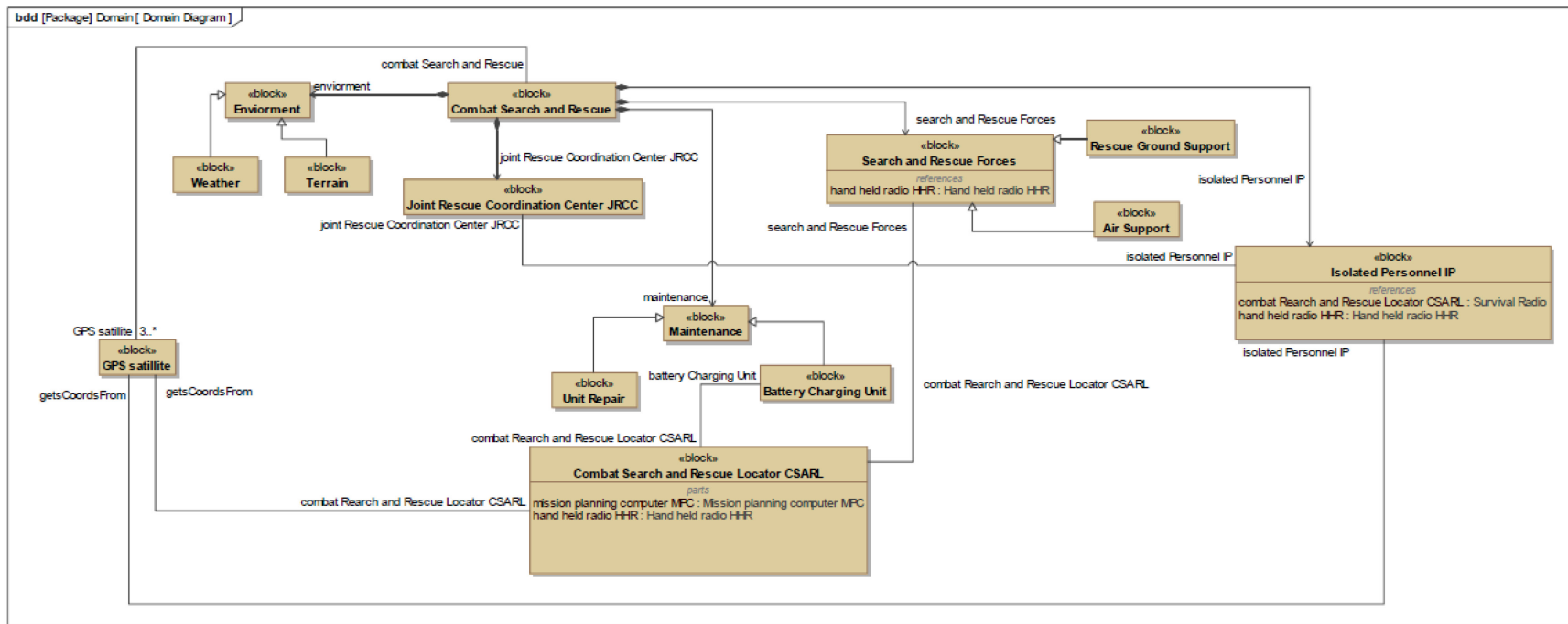
9. Would you recommend a test team use the “CAG” test plan?

| Yes                      | No                       |
|--------------------------|--------------------------|
| <input type="checkbox"/> | <input type="checkbox"/> |

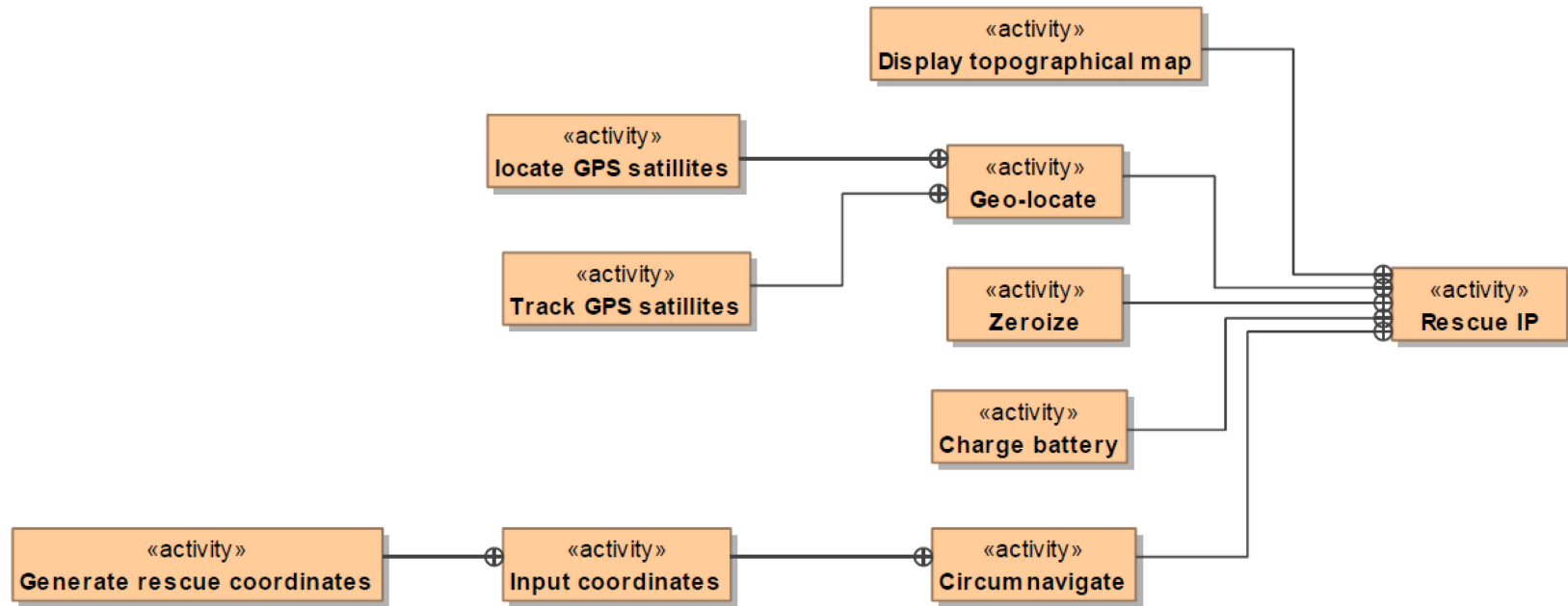
**Please explain your response:**

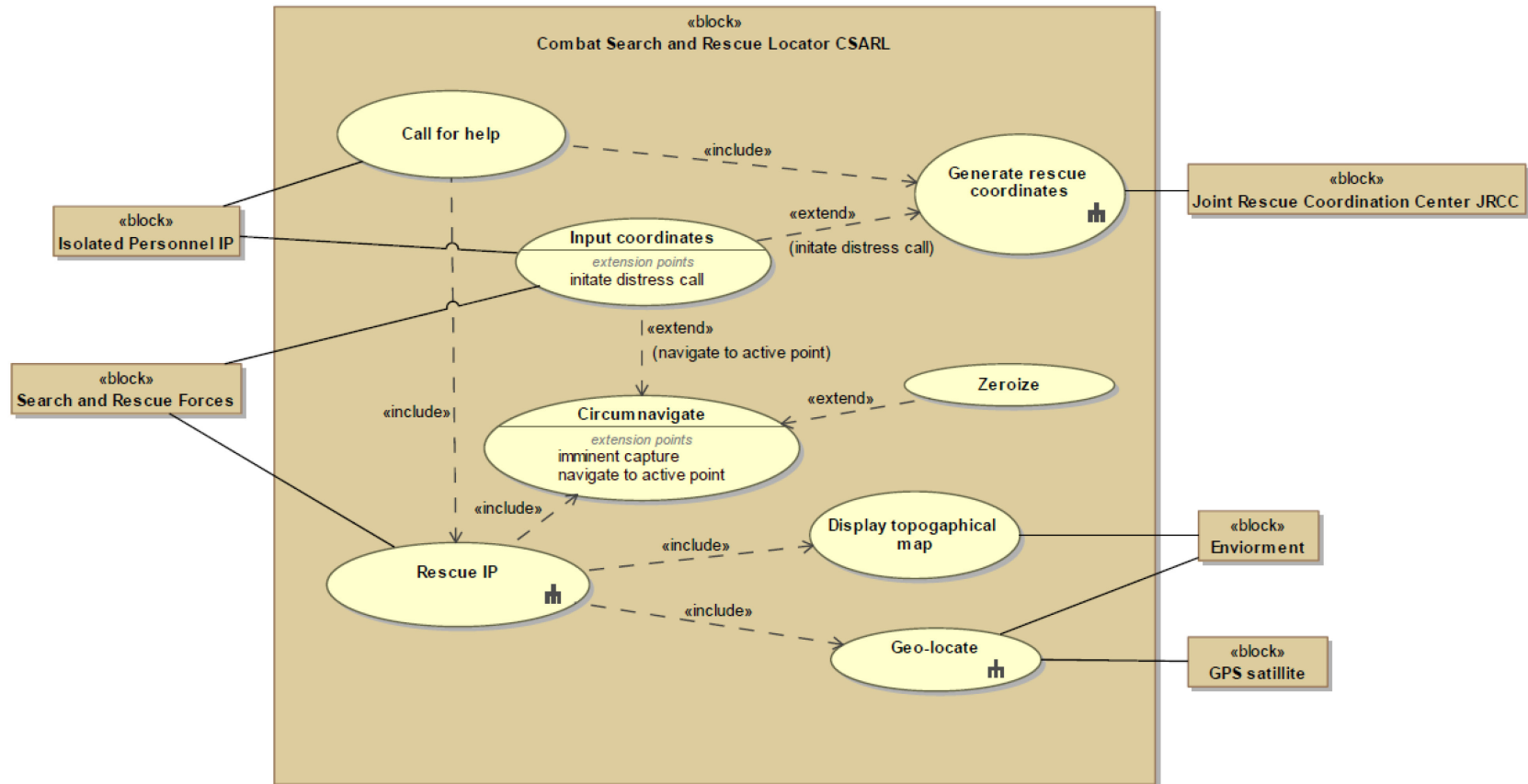
## APPENDIX B – COMBAT SEARCH AND RESCUE LOCATOR SYSML MODEL DIAGRAMS

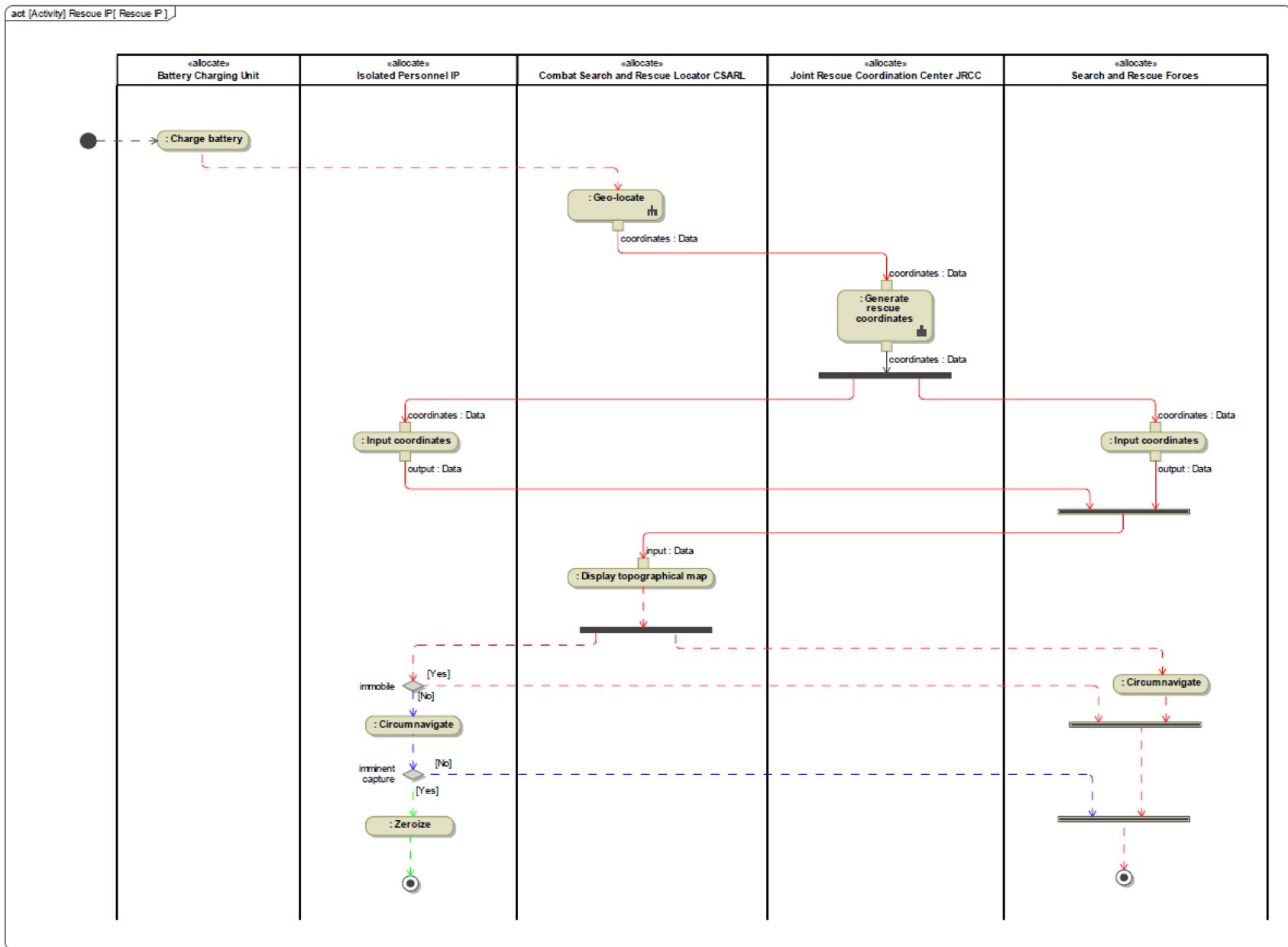




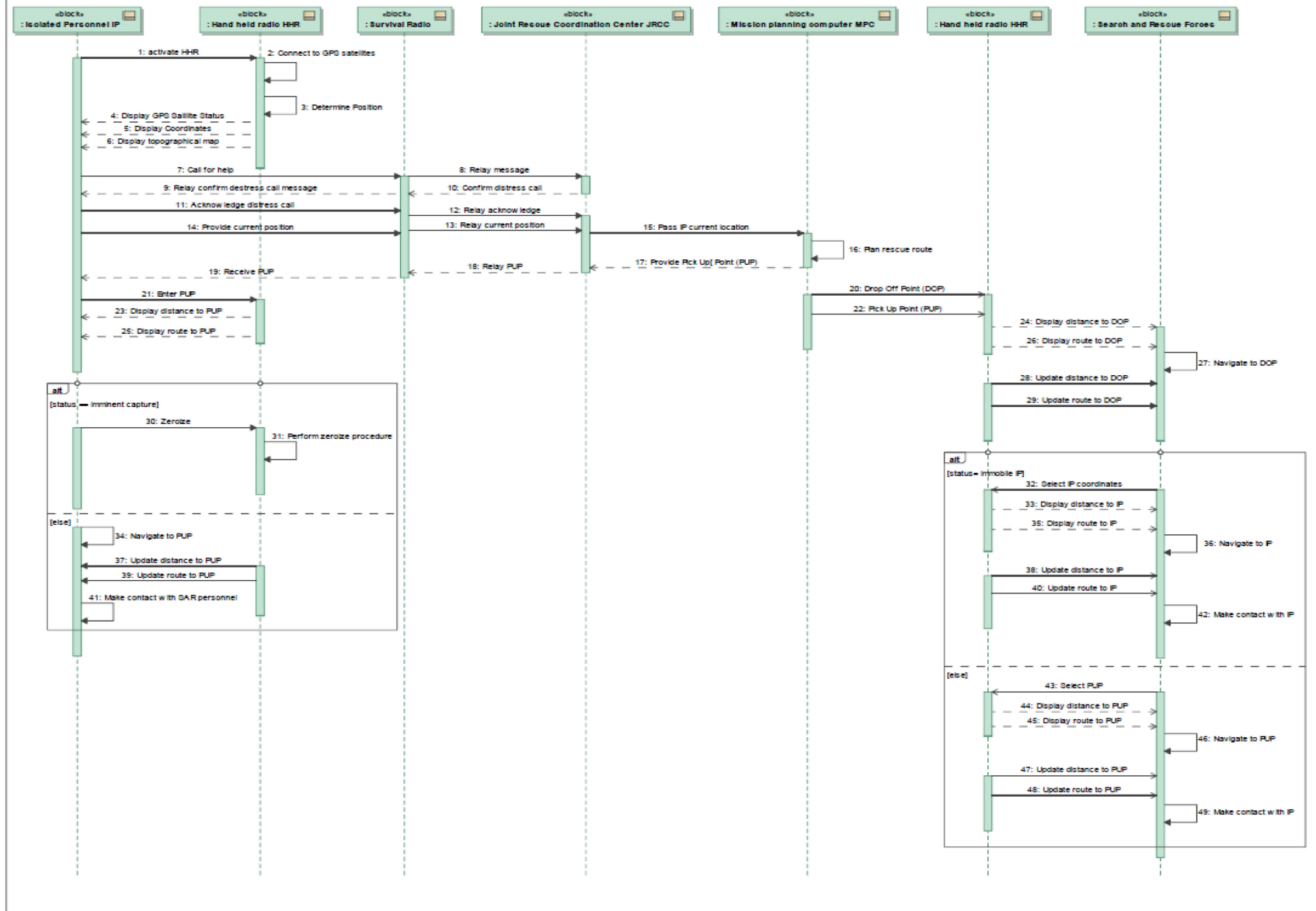
bdd [Package] Functional Decomposition [ Functional Decomposition Diagram ]

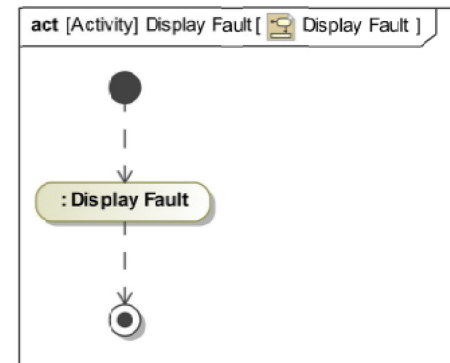
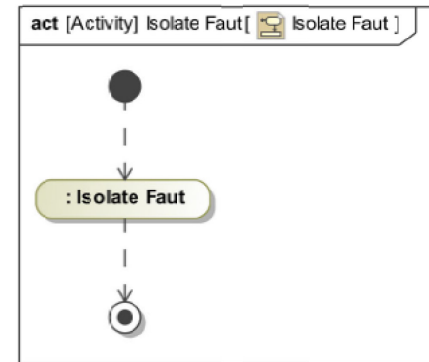
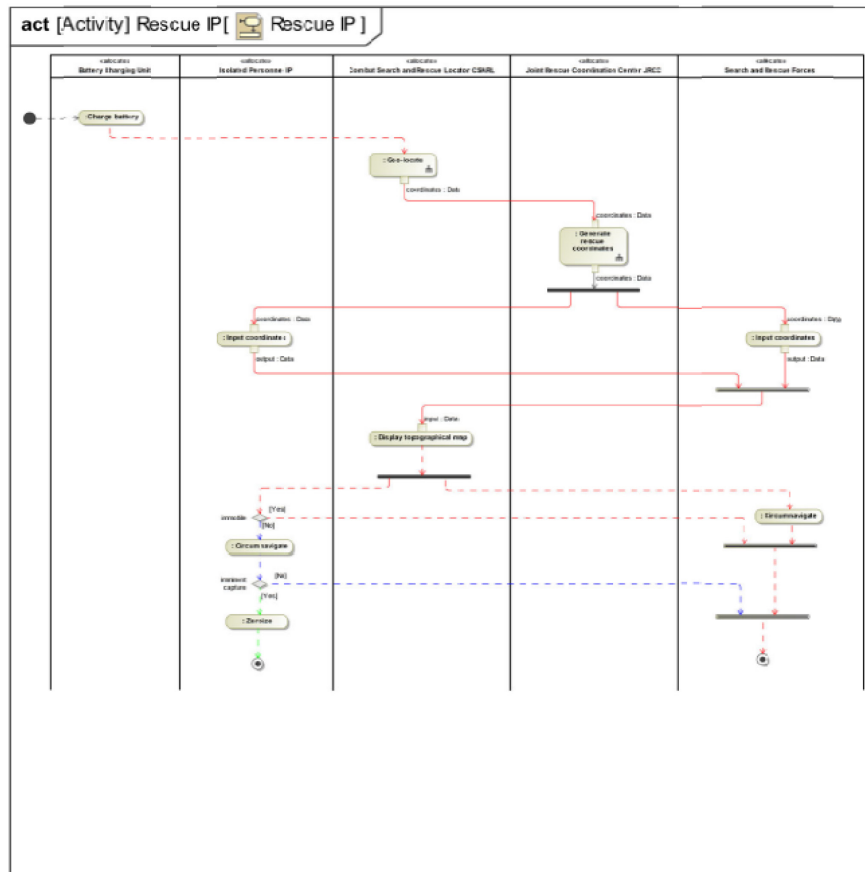
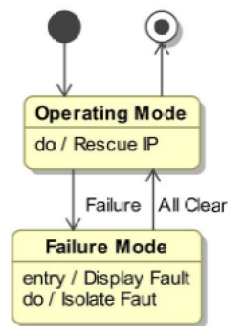






s.d [Interaction] Mission Sequence Diagram [ Mission Sequence Diagram ]







bdd [Package] COI [ COIs ]

«block»

**COI 1 - Can CSARL enable rescue team navigation to an isolated personnel location?**

«block»

**COI 2 -Can CSARL enable isolated personnel navigation to a rendezvous location?**

«block»

**COI 3 - Does CSARL support worldwide search and rescue operations?**

**bdd** [Block] COI 1 - Can CSARL enable rescue team navigation to an isolated personnel location? [ COI 1 - Can CSARL enable rescue team navigation to an isolated personnel location? ]

«block»  
**MOE 1.1 Support of IP Location**

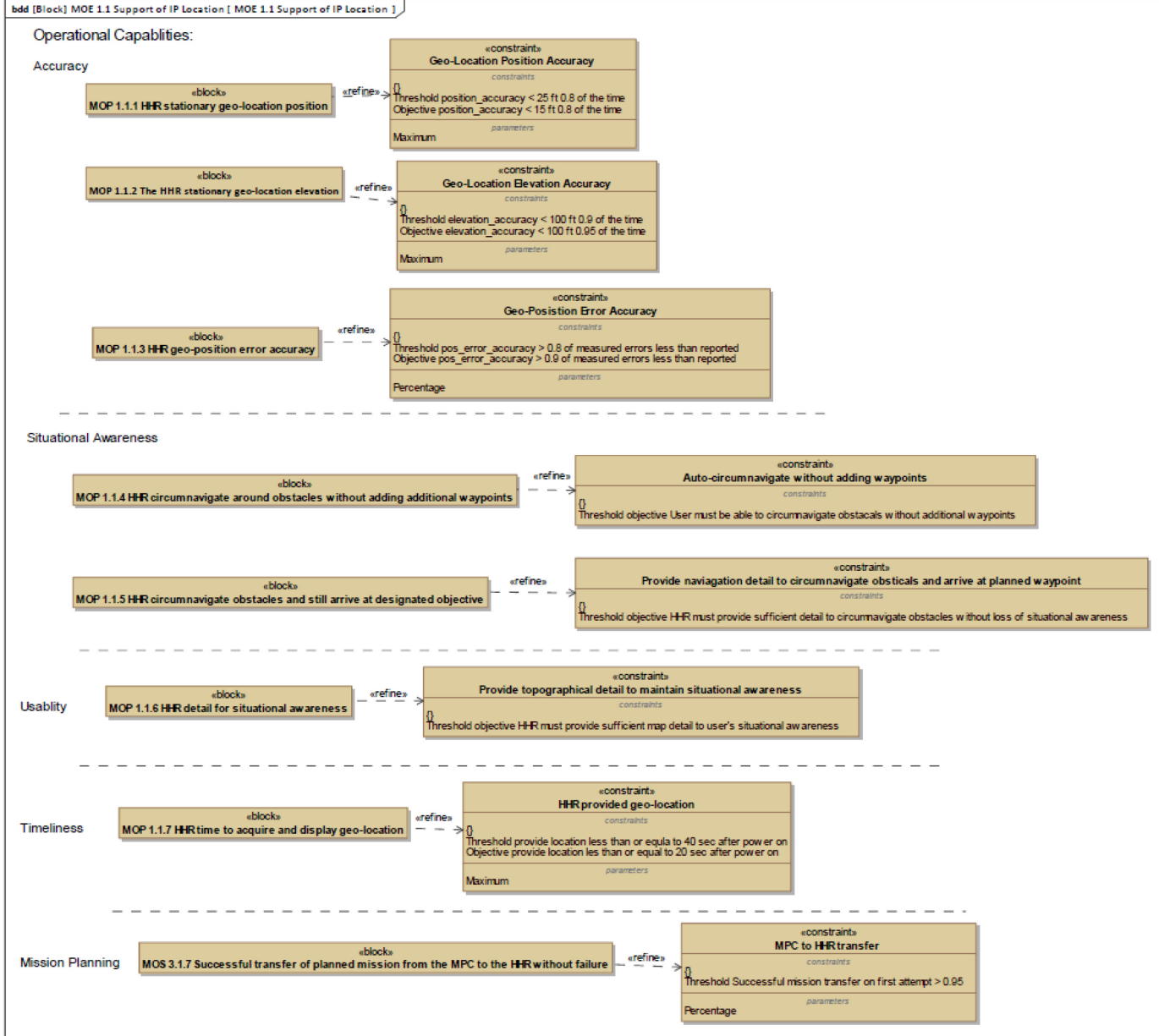
**bdd** [Block] COI 2 -Can CSARL enable isolated personnel navigation to a rendezvous location? [ COI 2 -Can CSARL enable isolated personnel navigation to a rendezvous location? ]

«block»  
**MOE 2.1 Support of Rendezvous Points Location**

«block»  
**MOE 2.2 HHR Cleared Successfully**

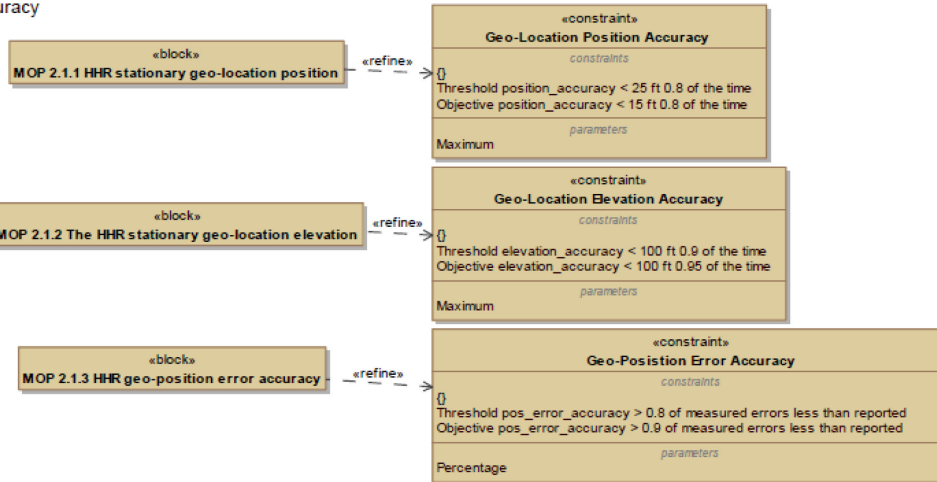
**bdd** [Block] COI 3 - Does CSARL support worldwide search and rescue operations? [ COI 3 - Does CSARL support worldwide search and rescue operations? ]

«block»  
**MOE 3.1 - CSARL Available For Operations**

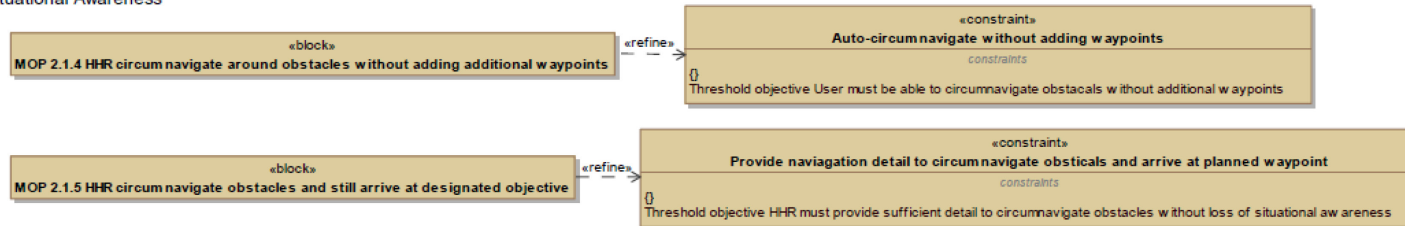


## Operational Capabilities

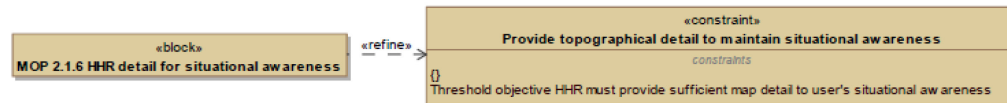
## Accuracy



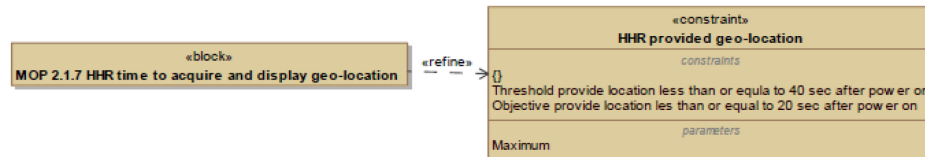
## Situational Awareness



## Useability



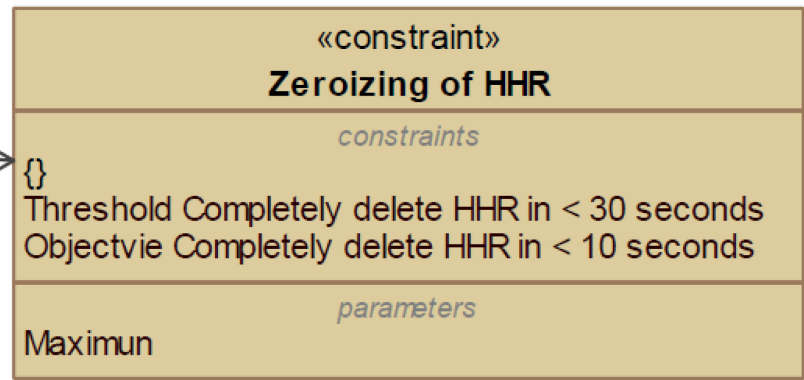
## Timeliness



bdd [Block] MOE 2.2 HHR Cleared Successfully[ MOE 2.2 HHR Cleared Successfully ]

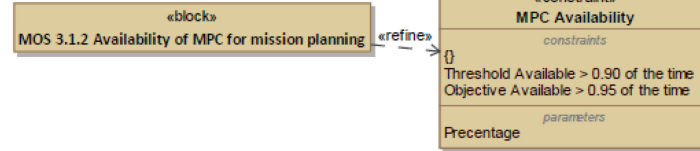
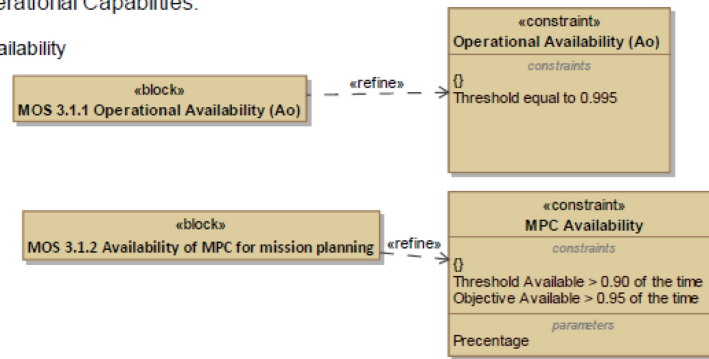
«block»  
**MOP 2.2.1 HHR Cleared Successfully**

«refine»

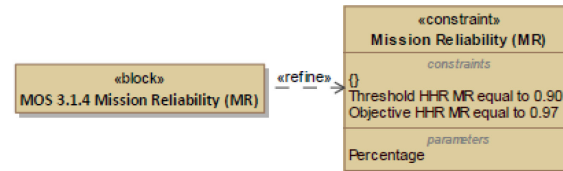
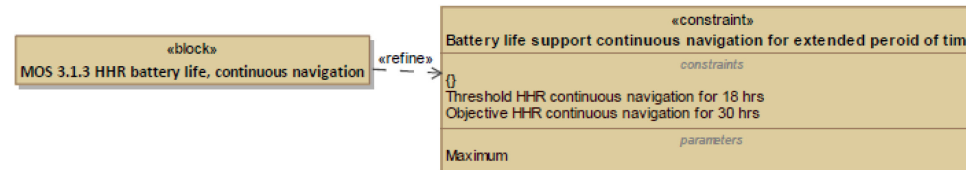


## Operational Capabilities:

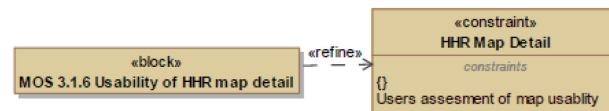
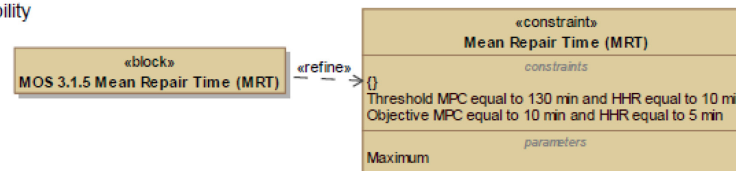
## Availability



## Reliability

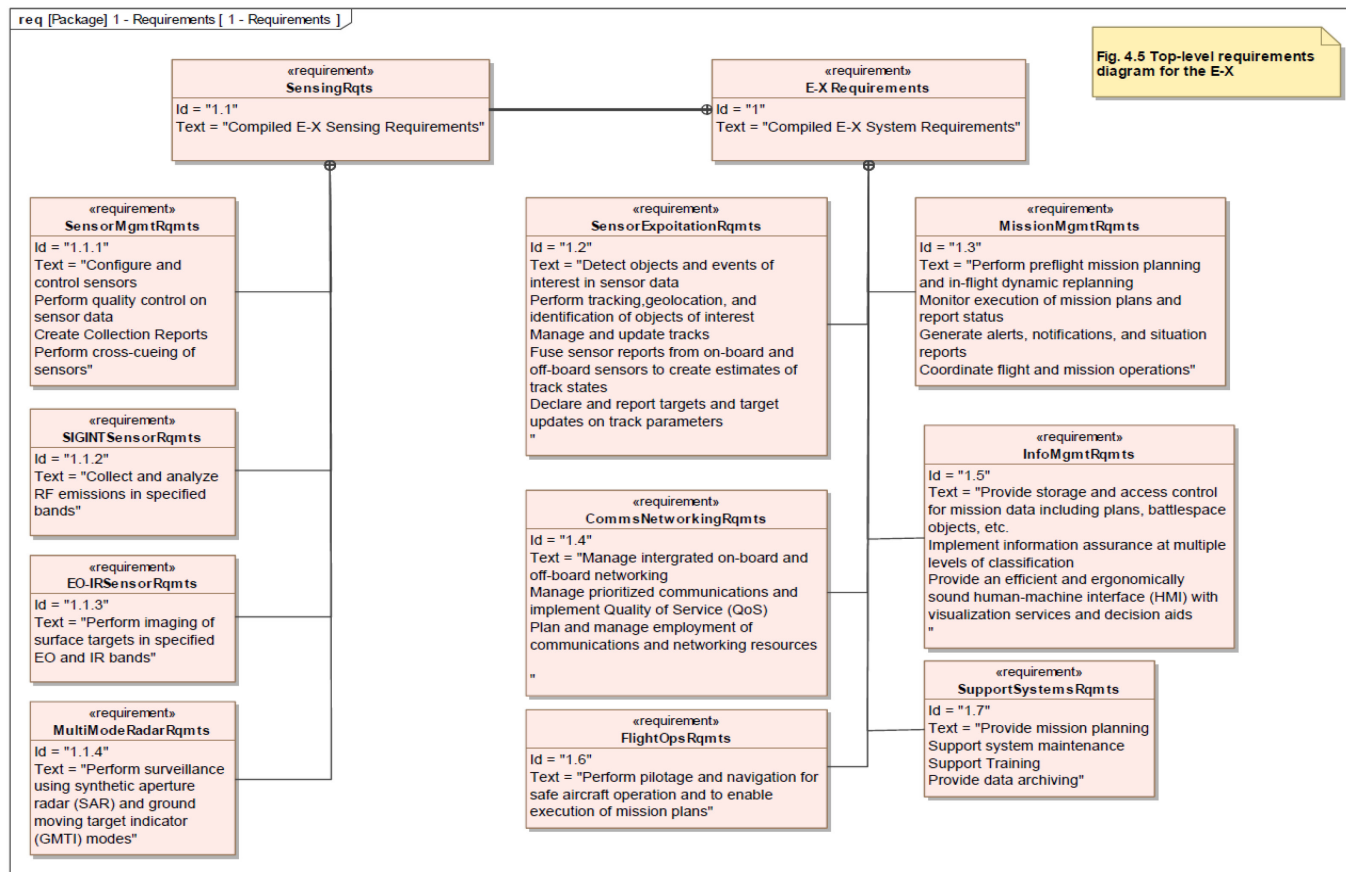


## Maintainability



## APPENDIX C – E-X AIRBORNE MULTI-SENSOR PLATFORM SYSML MODEL DIAGRAMS

The following diagrams have been recreated as they appear in the Effective Model-Based Systems Engineering textbook by Borky and Bradley.



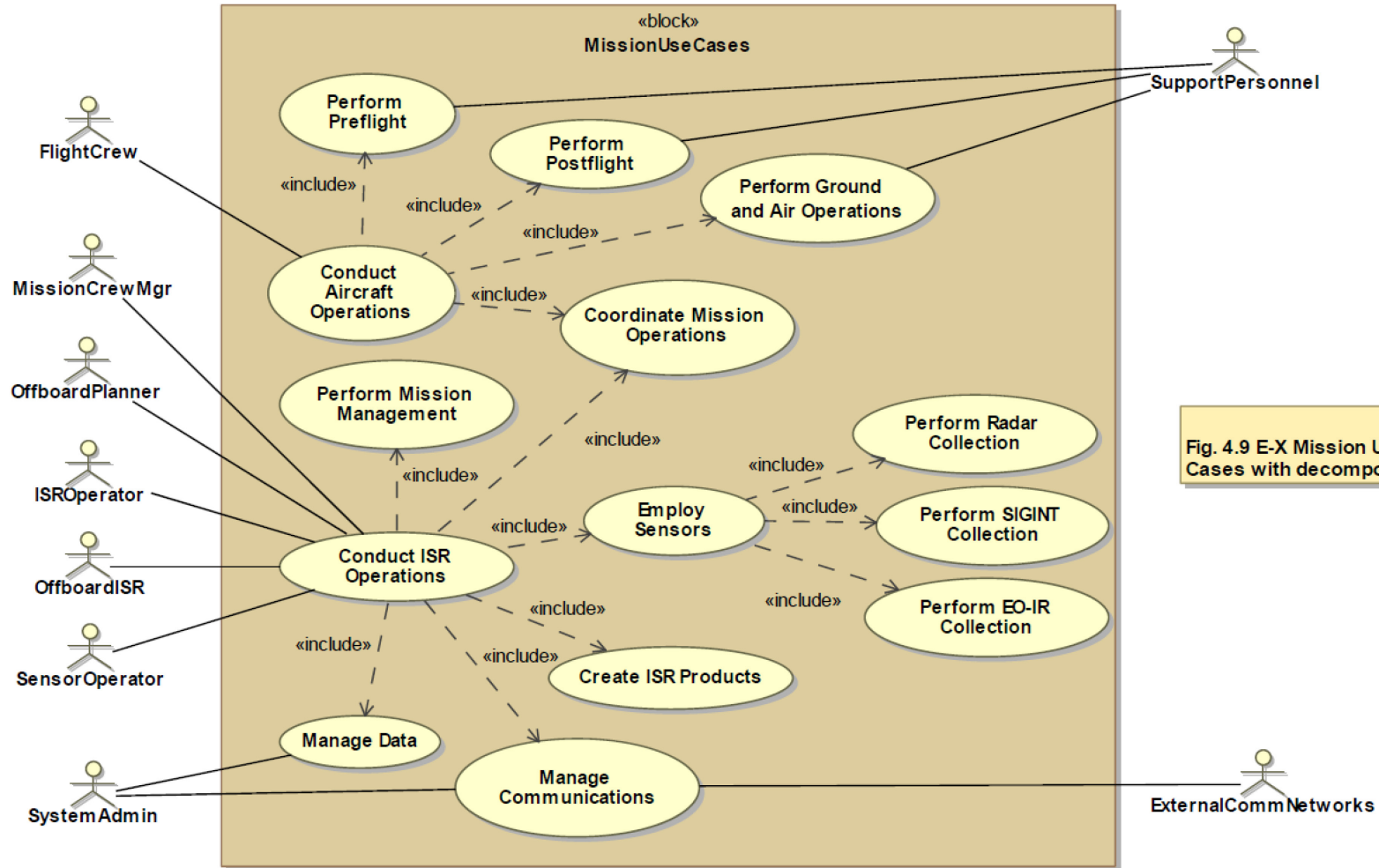


Fig. 4.9 E-X Mission Use Cases with decomposition





act [Activity] Create ISR Product [ Create ISR Product ]

Fig. 4.11 Scenario Activity Diagram for the Create ISR Report Use Case

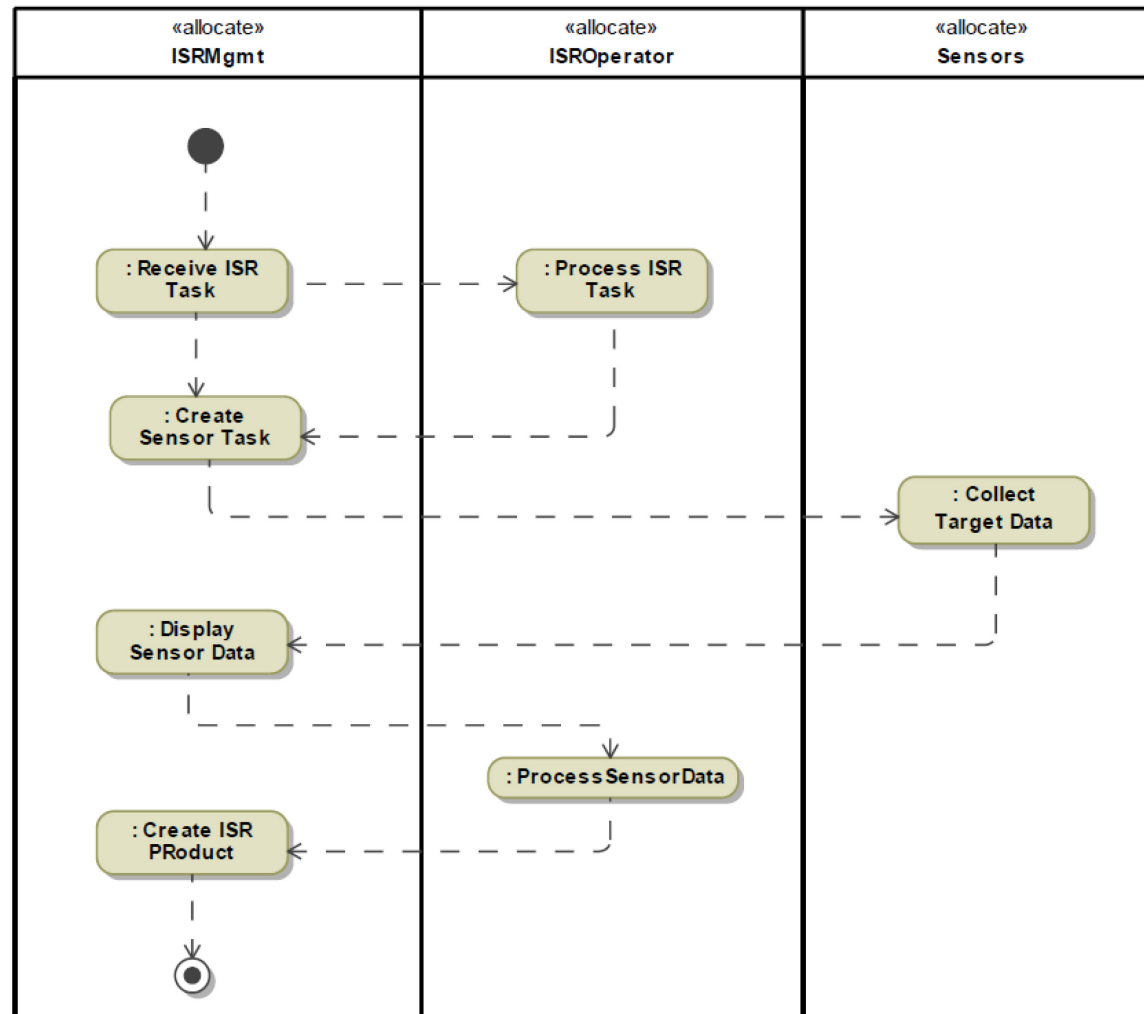


Fig C.15 E-X Operational Context

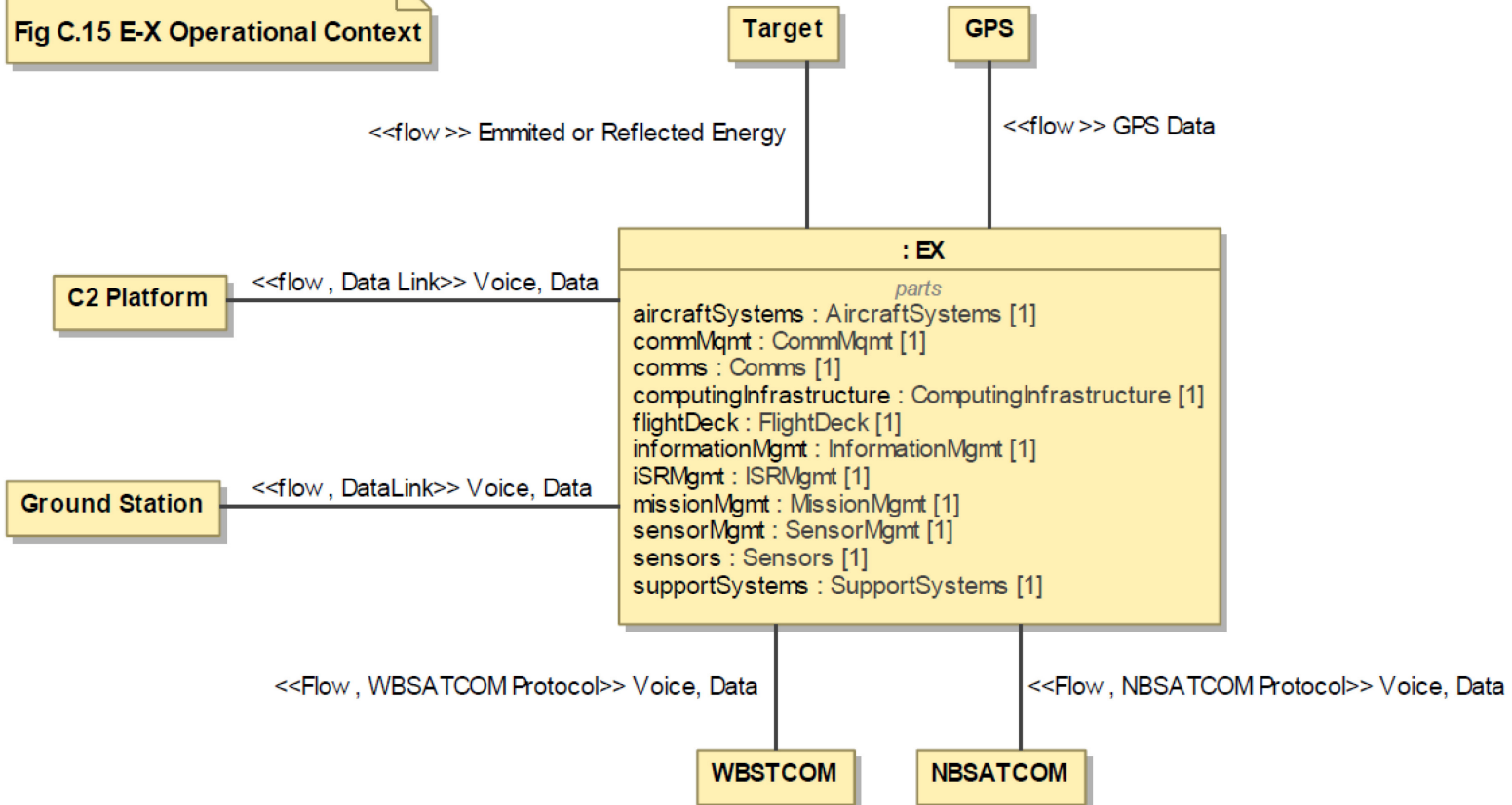
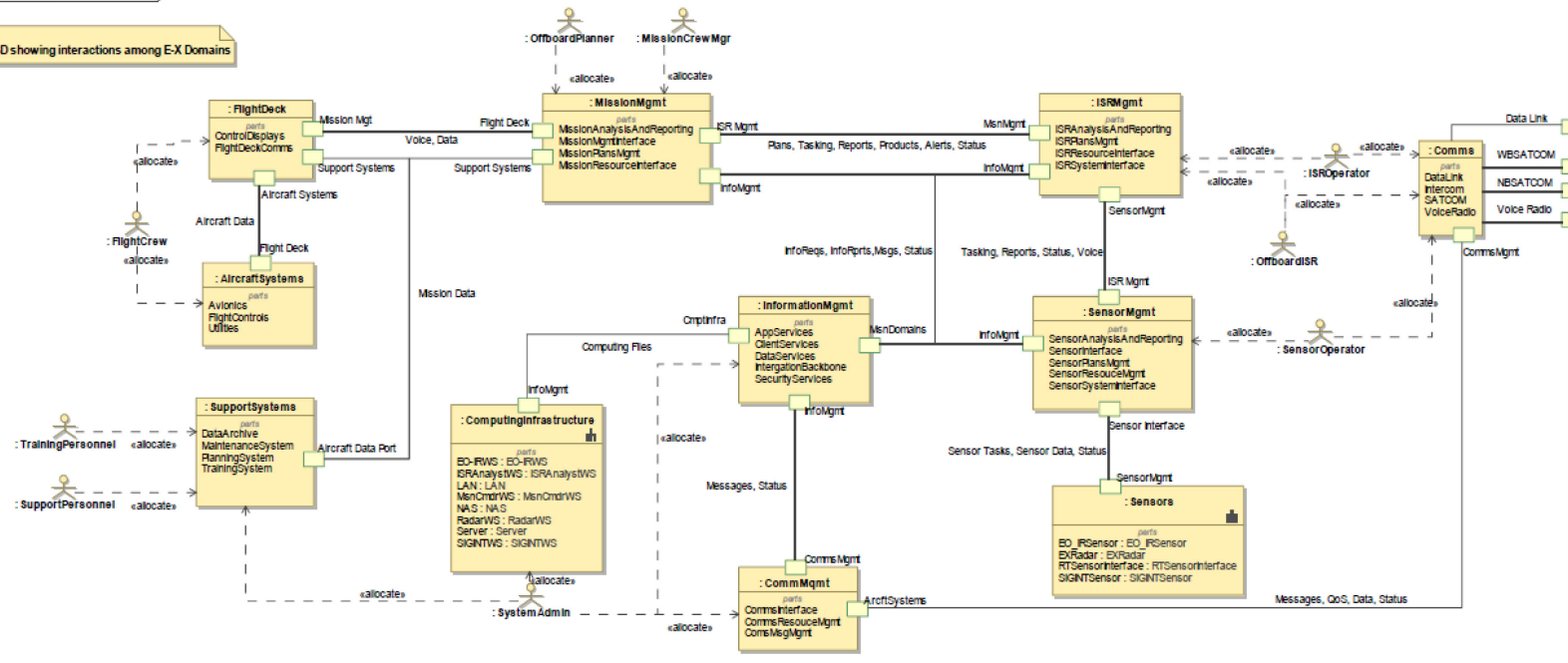
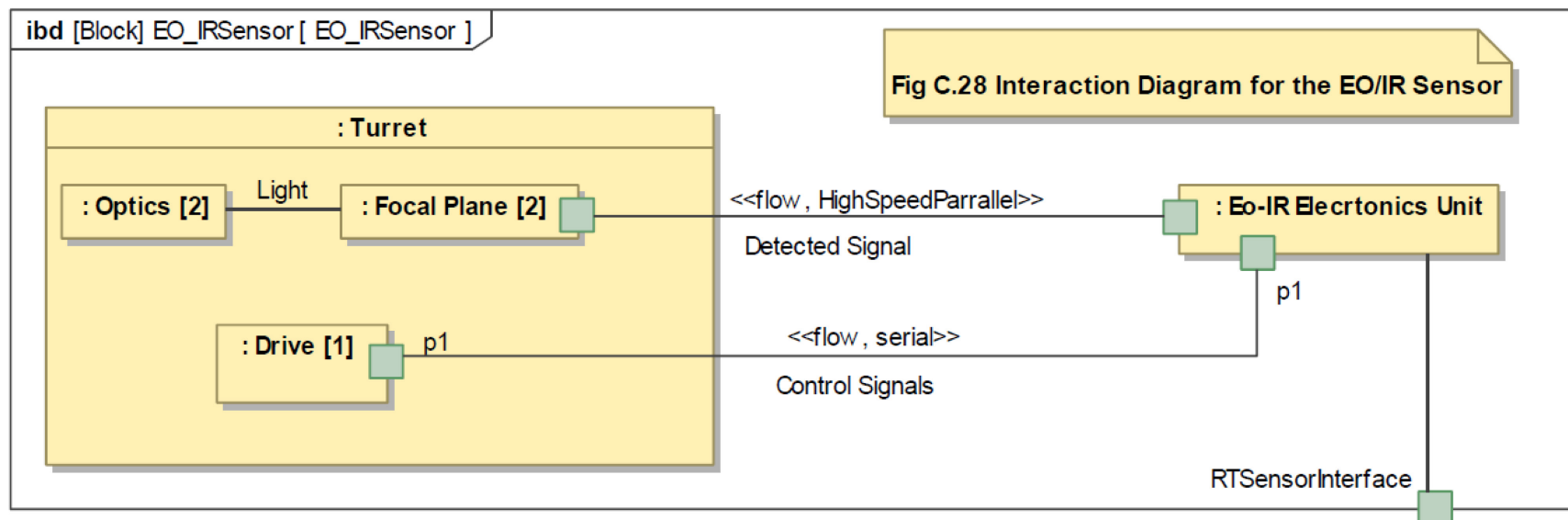
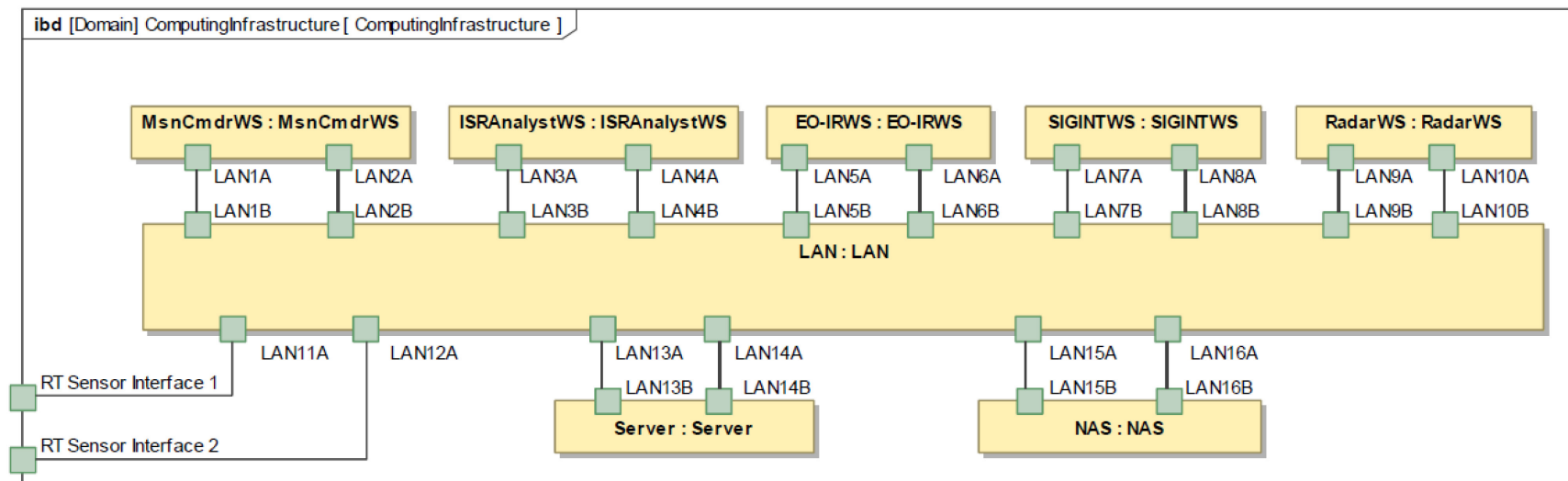
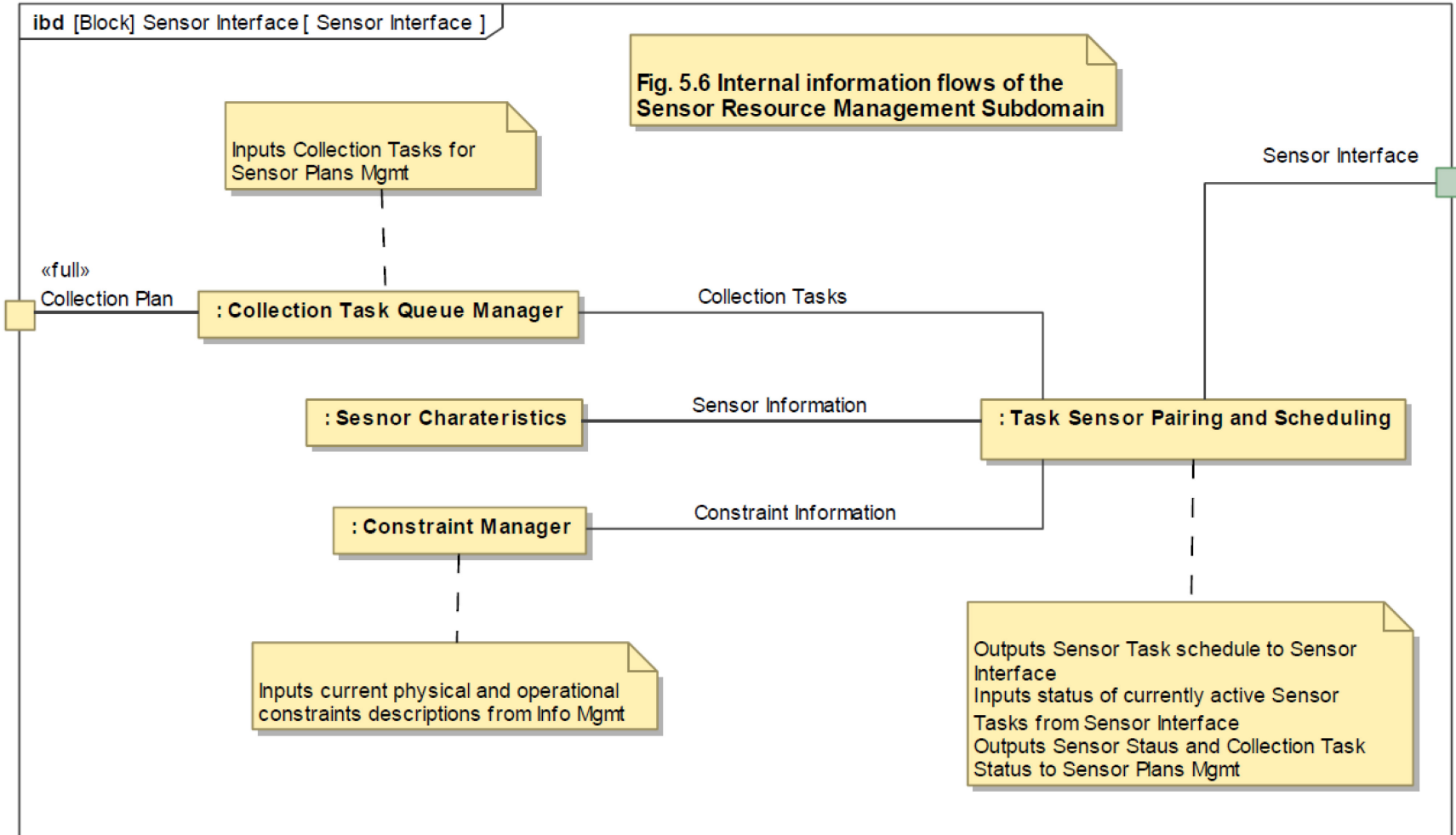
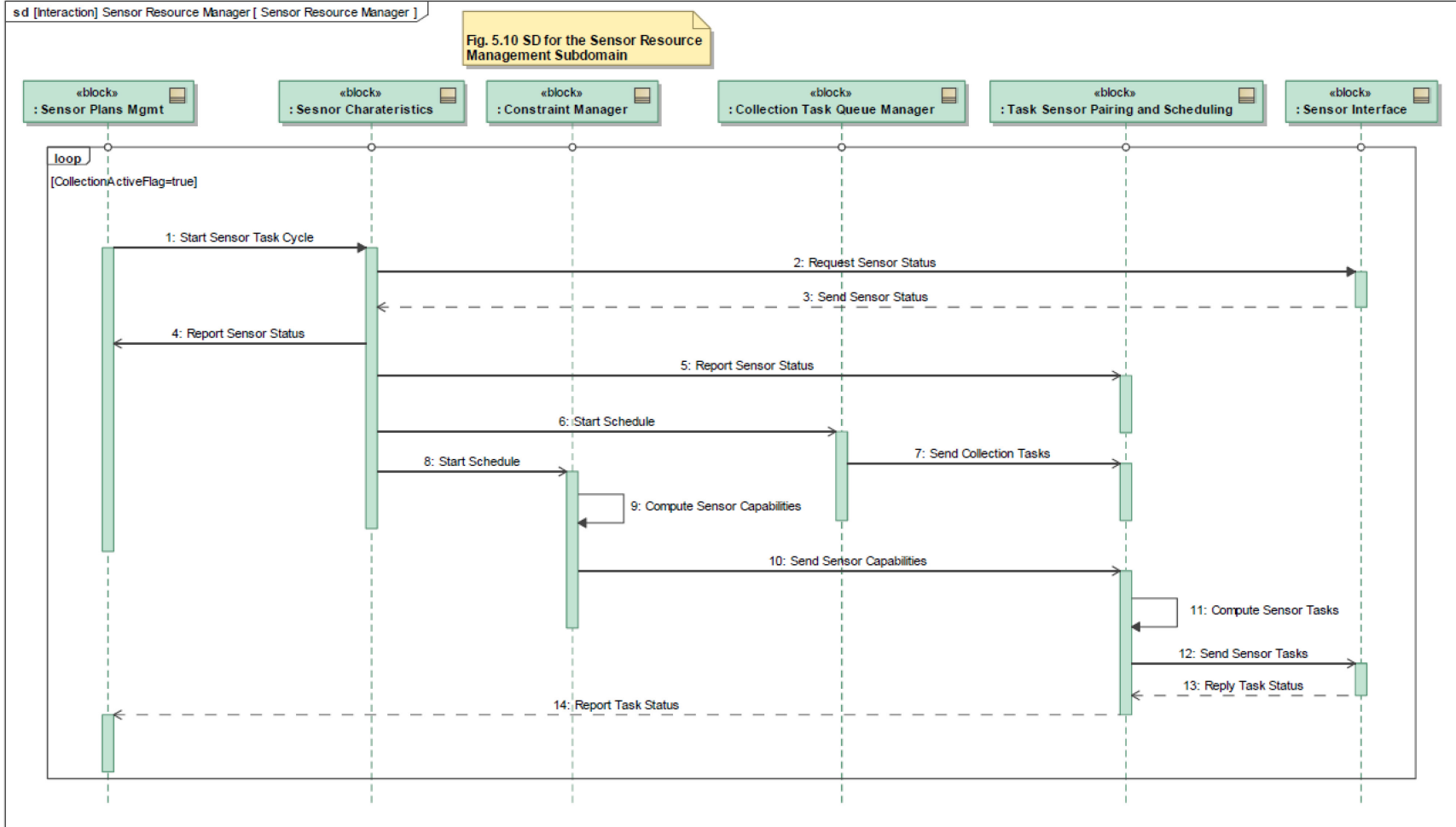


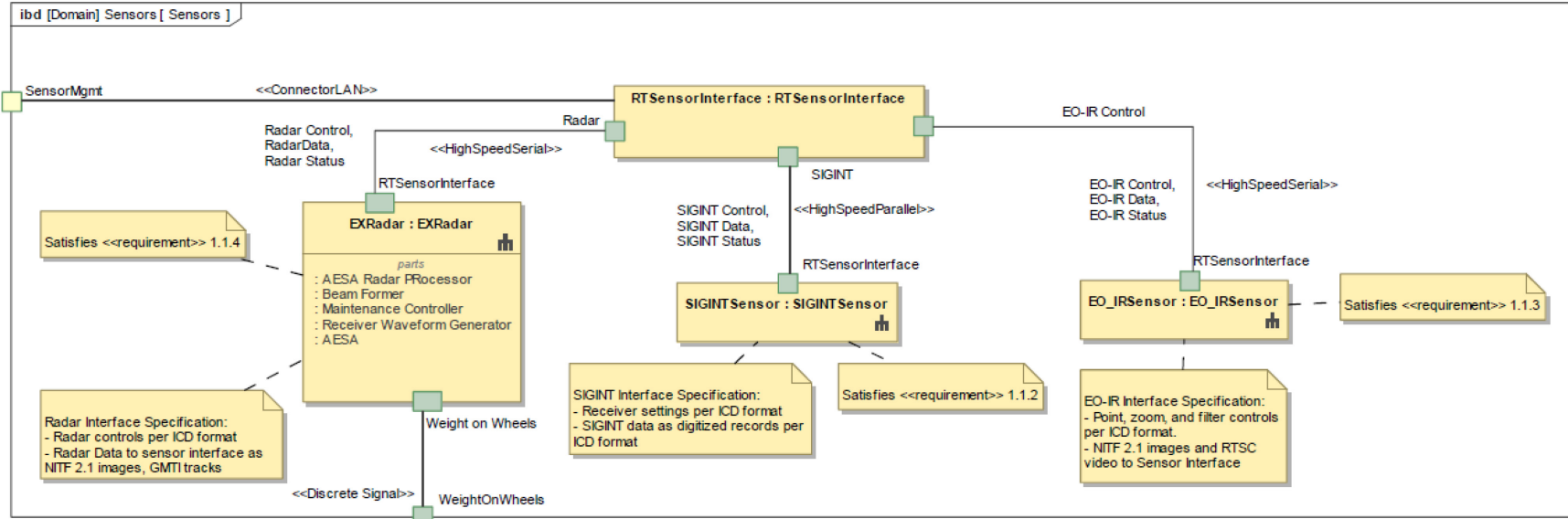
Fig. 4.7 IBD showing interactions among E-X Domains



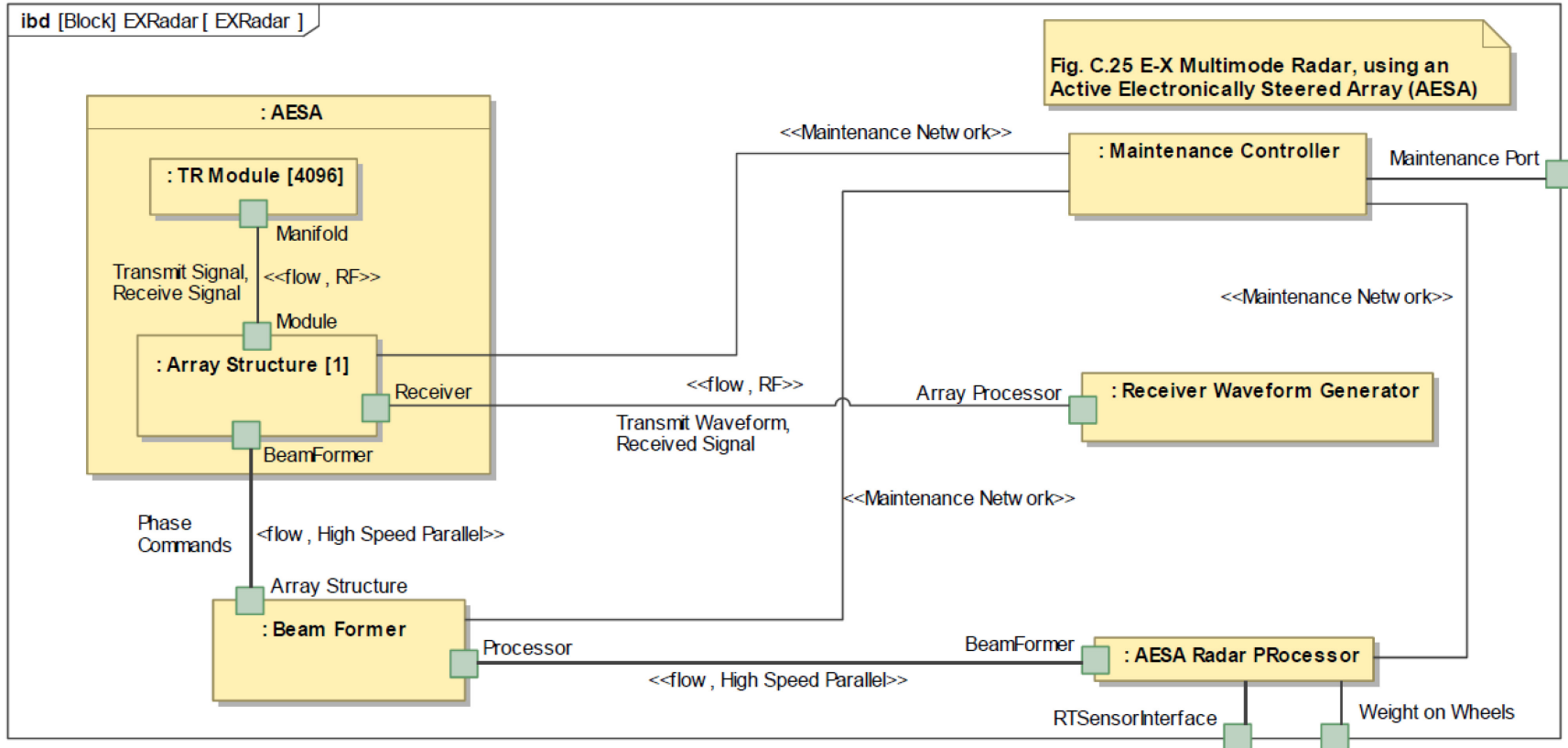


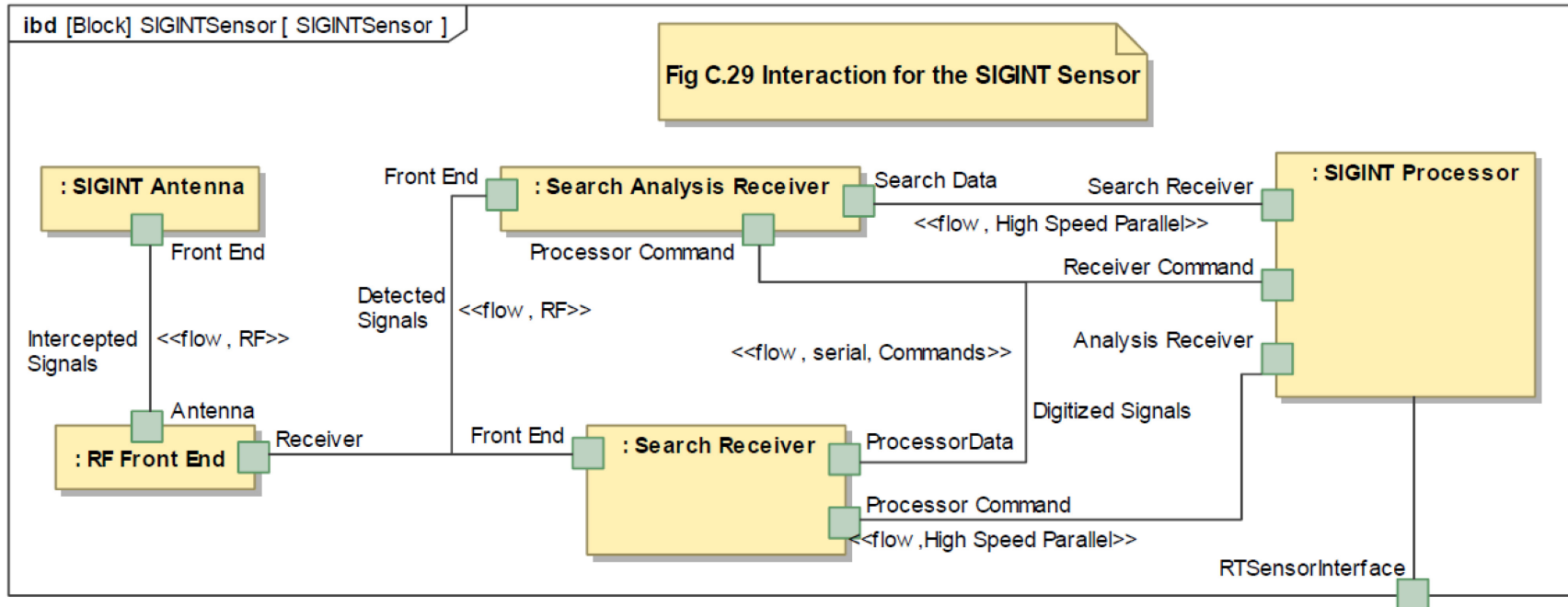


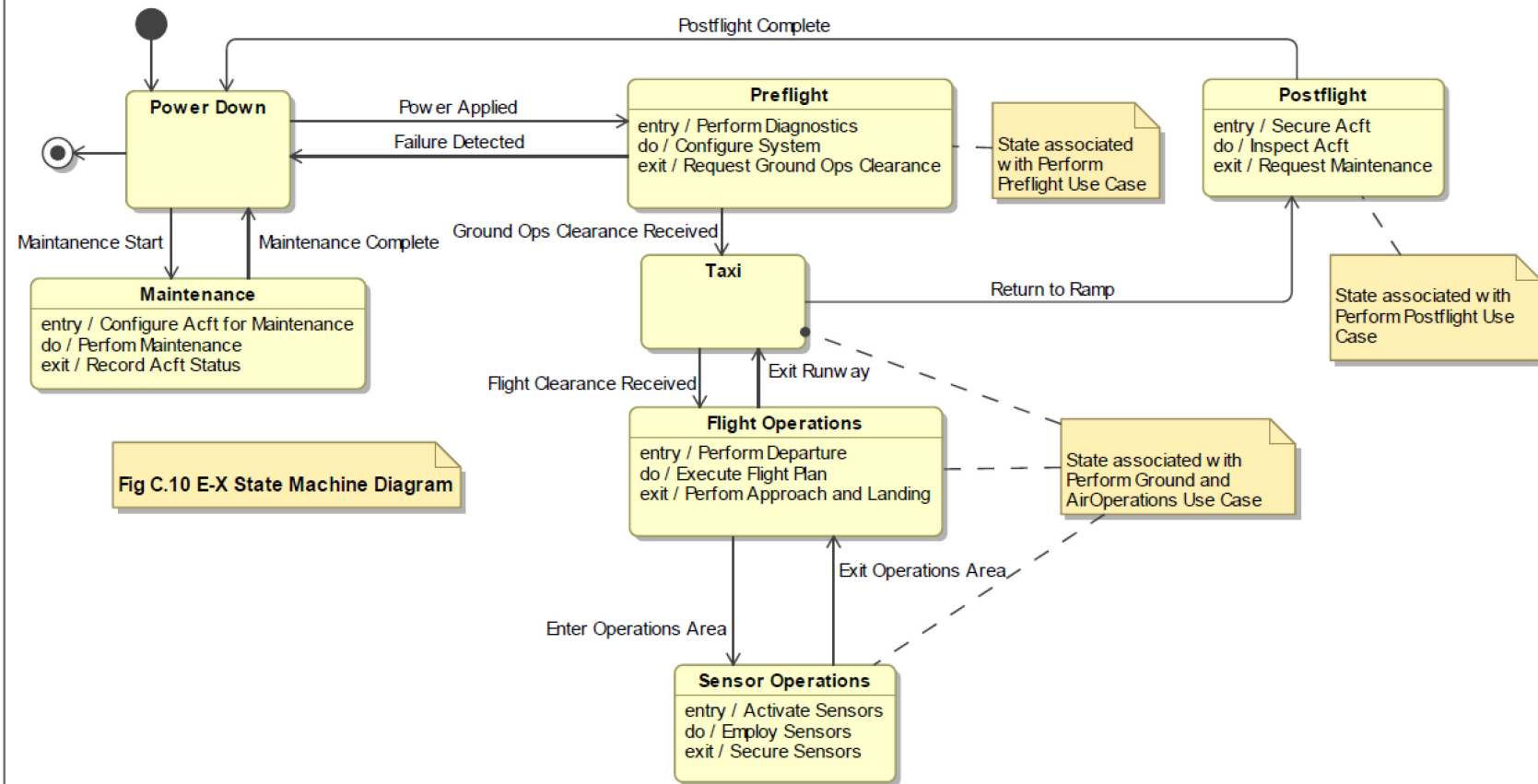




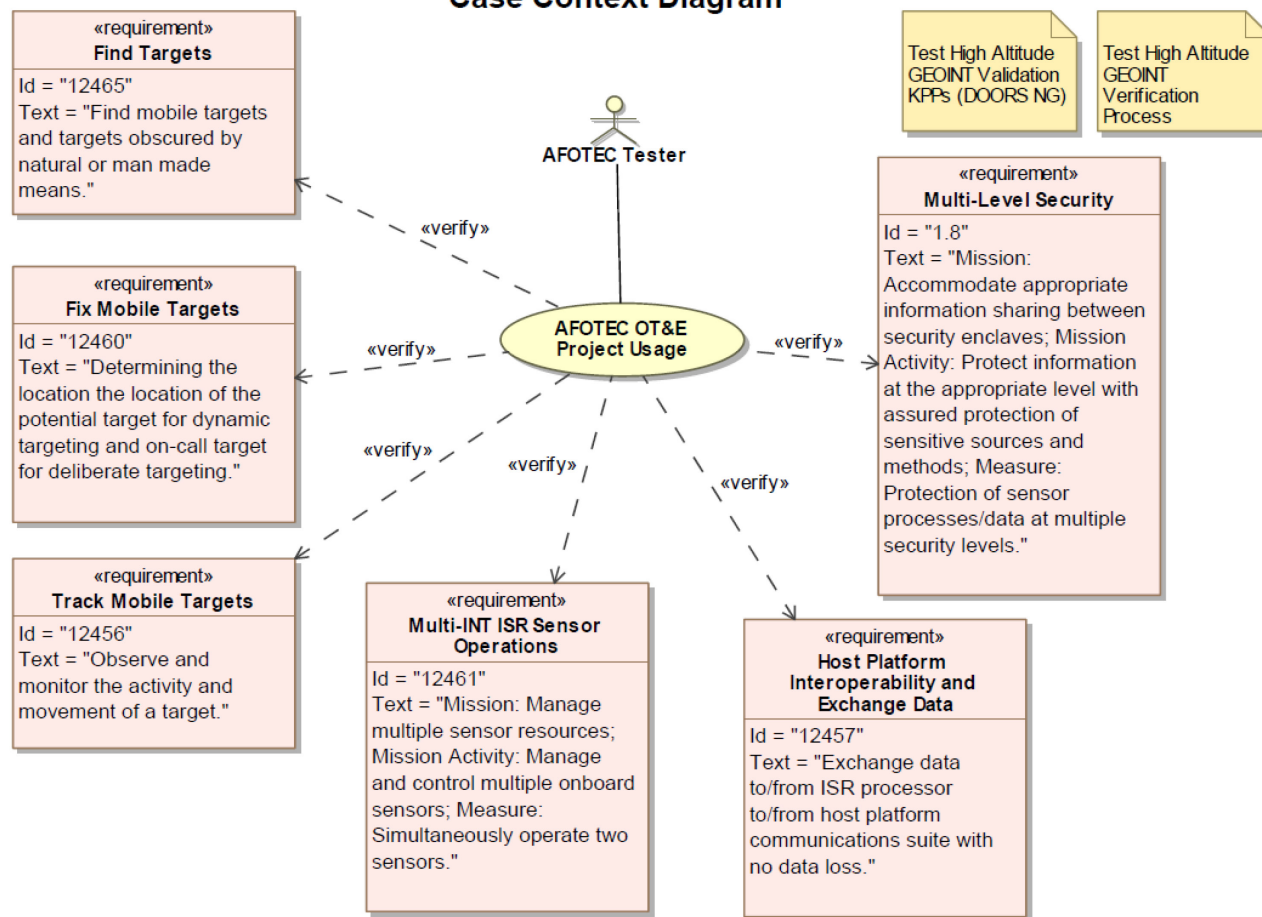








## Test High Altitude GEOINT Use Case Context Diagram



«Critical Operation Issue»  
«block»

**COI 1 - Can the E-X Sensor Platform perform wide area surveillance ?**

«Critical Operation Issue»  
«block»

**COI 2 - Can the E-X Platform collect signal data to support Combatant Commander taskings?**

«Critical Operation Issue»  
«block»

**COI 3 - Can the E-X Platform support exploitation and dissemination of ISR information to end users?**

«Critical Operation Issue»  
«block»

**COI 4 - Is E-X Airborne Sensor Platform employable to support the TACS?**

«Critical Operation Issue»  
«block»

**COI 5 - Is E-X Airborne Sensor Platform sustainable to support the TACS?**

**bdd** [Block] COI 1 - Can the E-X Sensor Platform perform wide area surveillance ? [ COI 1 - Can the E-X Sensor Platform perform wide area surveillance ? ]

«block»

**MOE 1.1 Continuous detection, identification, and tracking of airborne targets**

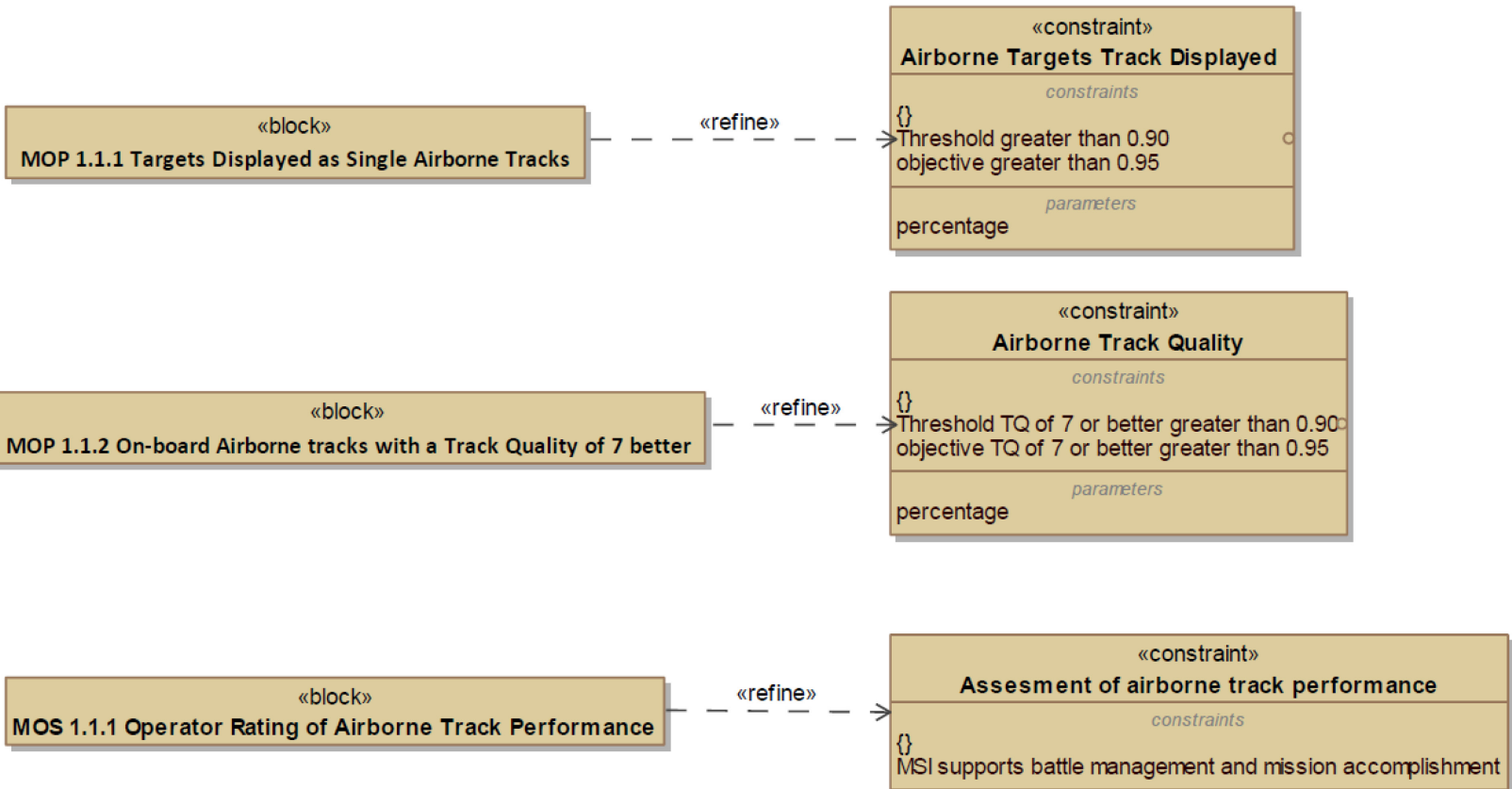
«block»

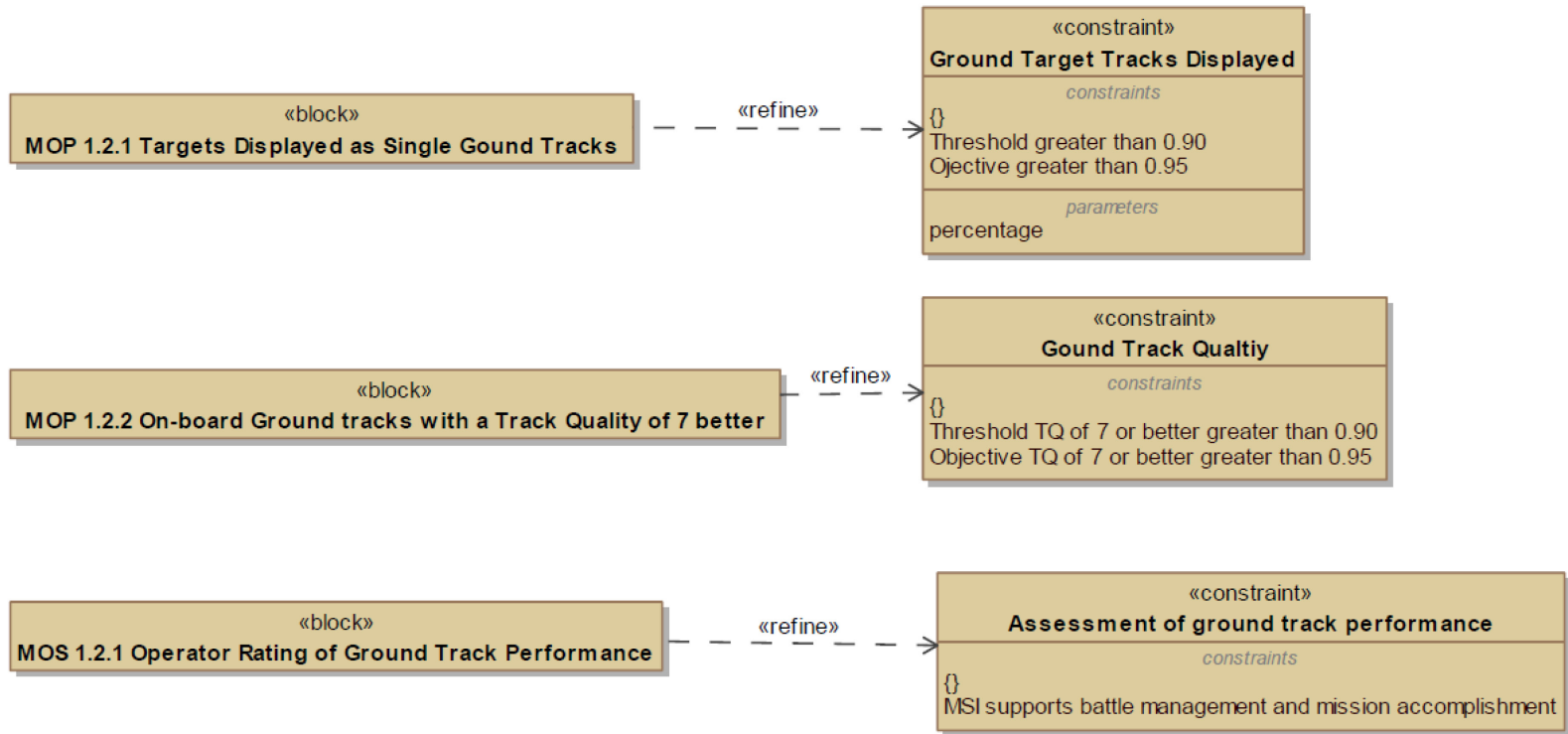
**MOE 1.2 Continuous detection, identification, and tracking of ground targets**

«block»

**MOE 1.3 Continuous detection, identification, and tracking of maritime targets**

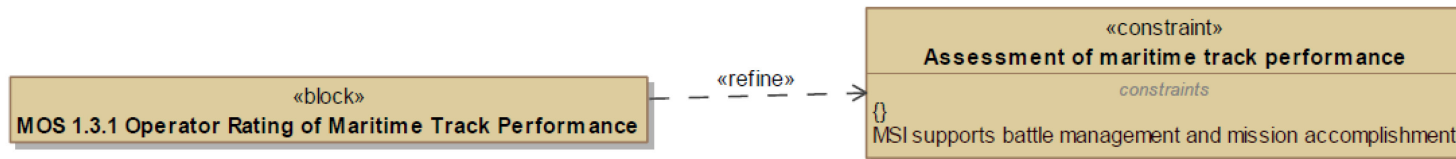
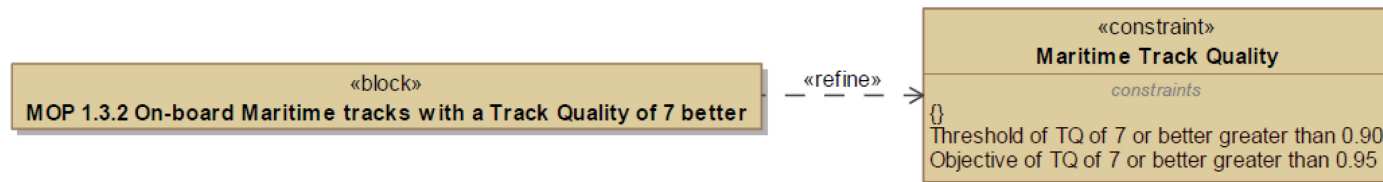
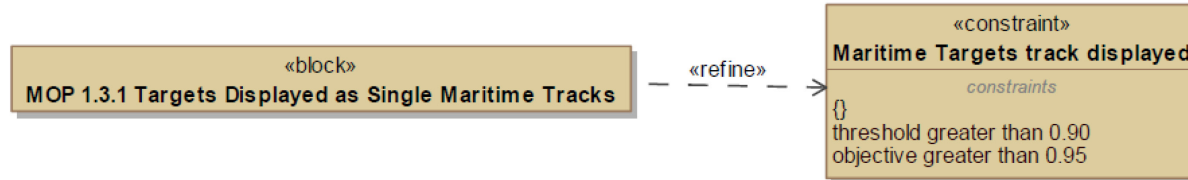
**bdd** [Block] MOE 1.1 Continuous detection, identification, and tracking of airborne targets [ MOE 1.1 ]



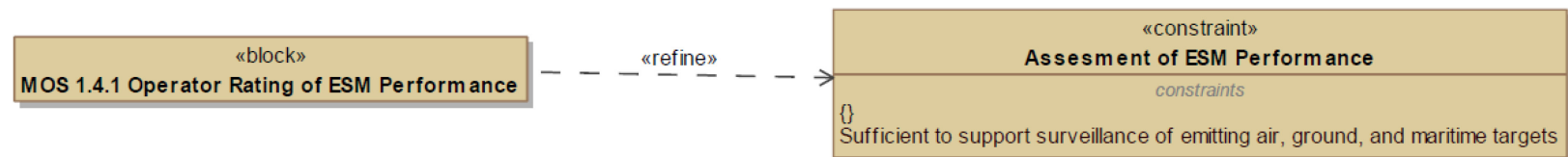
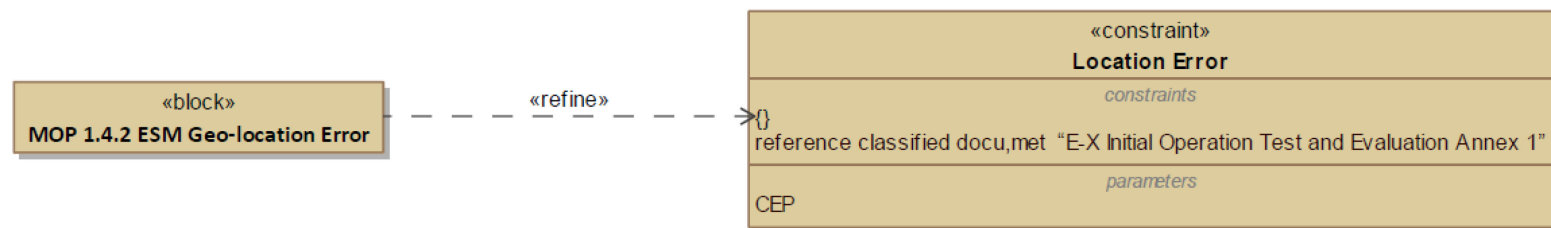
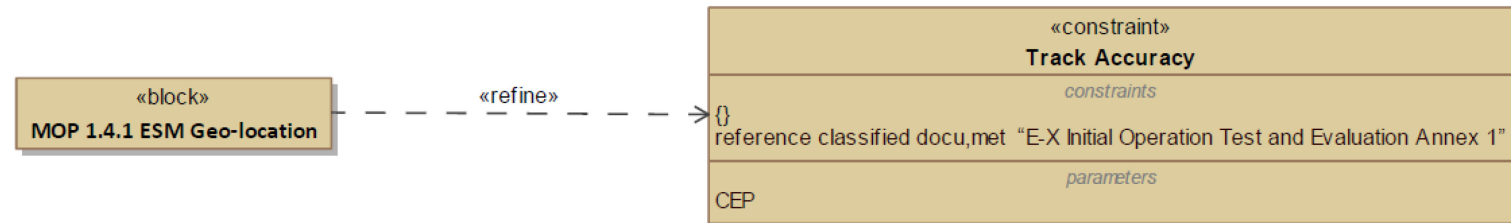




**bdd** [Block] MOE 1.3 Continuous detection, identification, and tracking of maritime targets [ MOE 1.3 ]



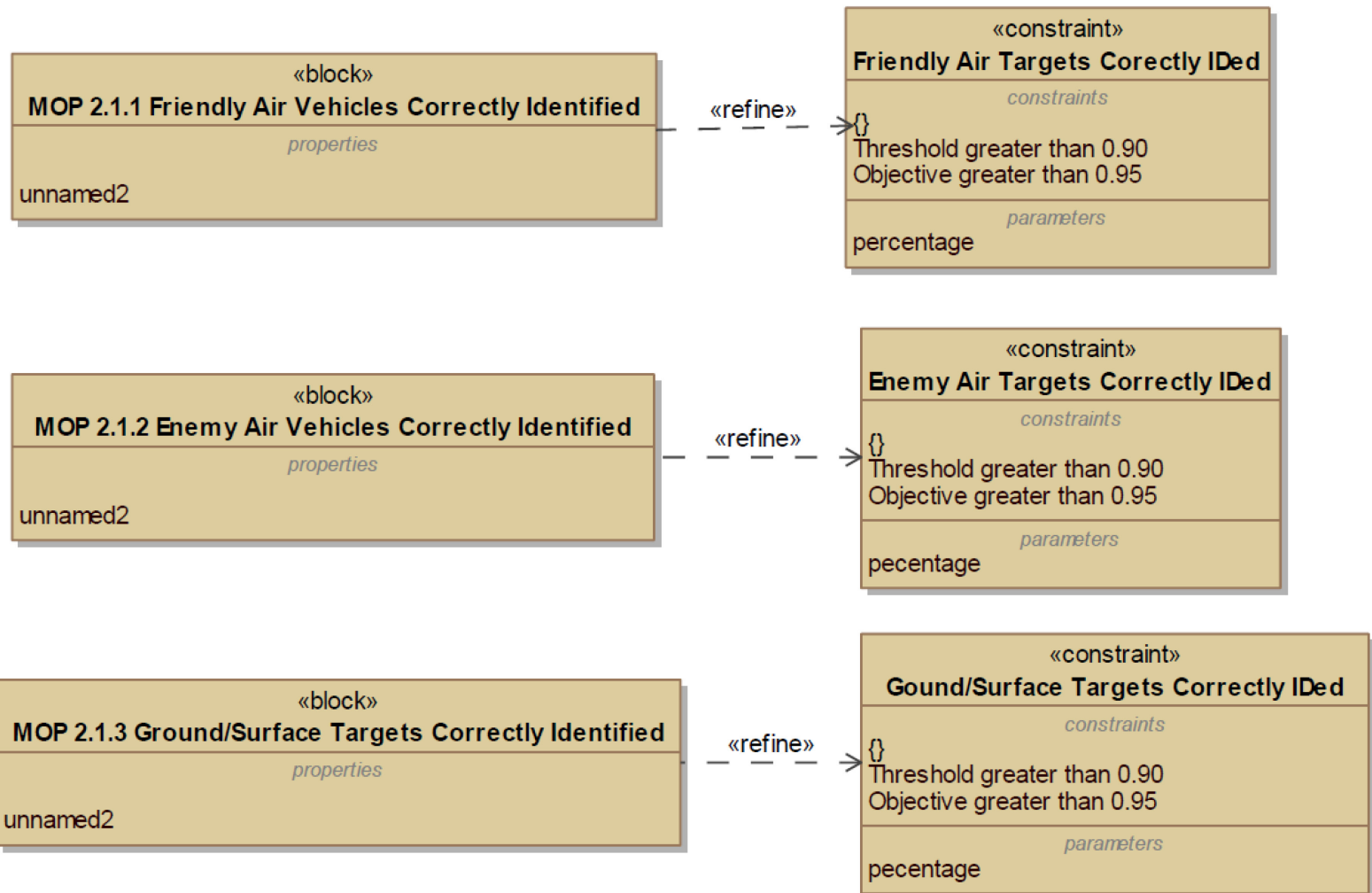
bdd [Block] MOE 1.4 Continuous detection, identification, and tracking of ESM targets [ MOE 1.4 ]



bdd [Block] COI 2 - Can the E-X Platform collect signal data to support Combatant Commander taskings? [ COI 2 - Can the E-X Platform collect signal data to support Combatant Commander taskings? ]

«block»  
**MOE2.1 Correct identification of targets for tactical picture**

«block»  
**MOE 2.2 Maintain communication and distribution of the common tactical picture**

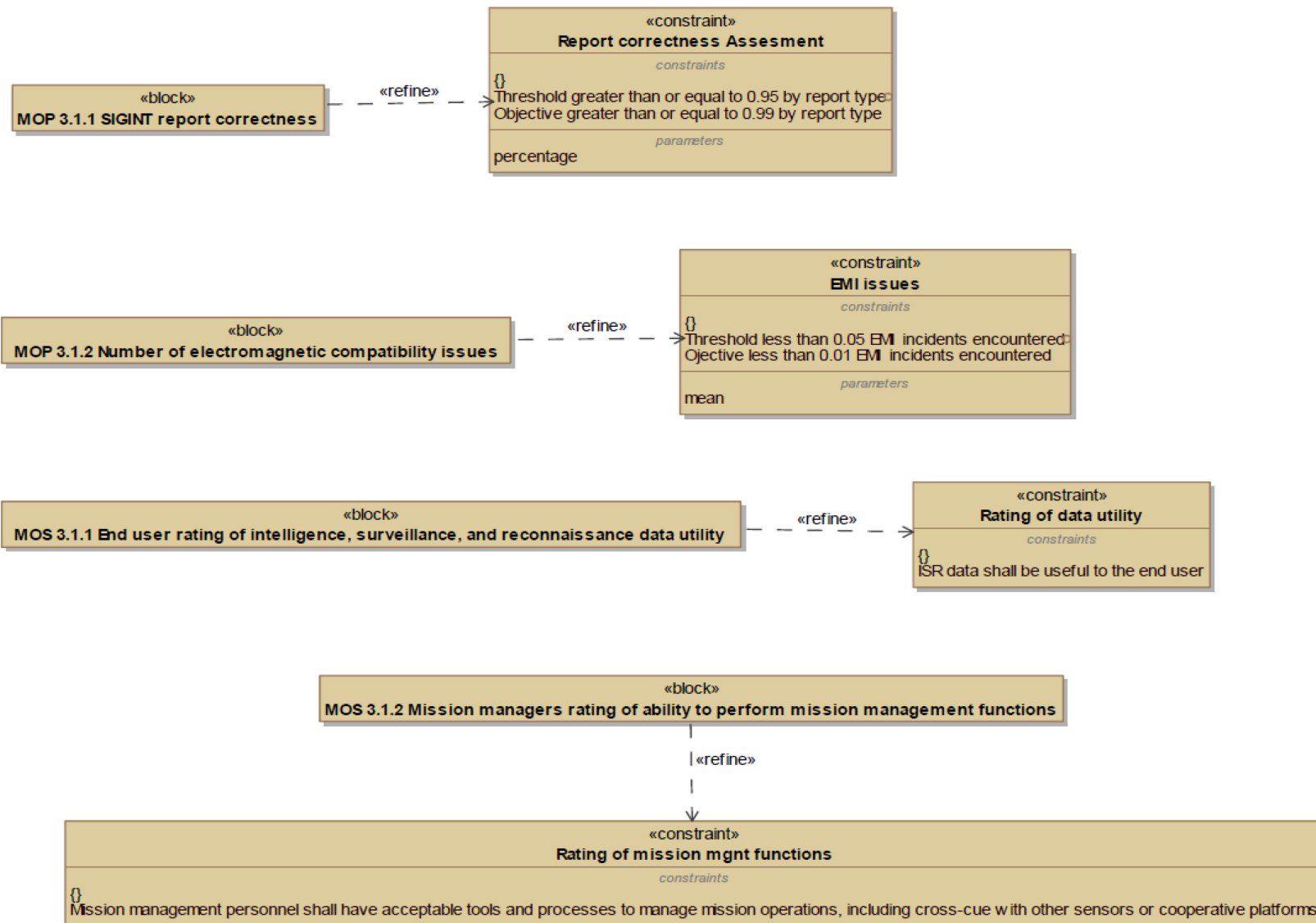


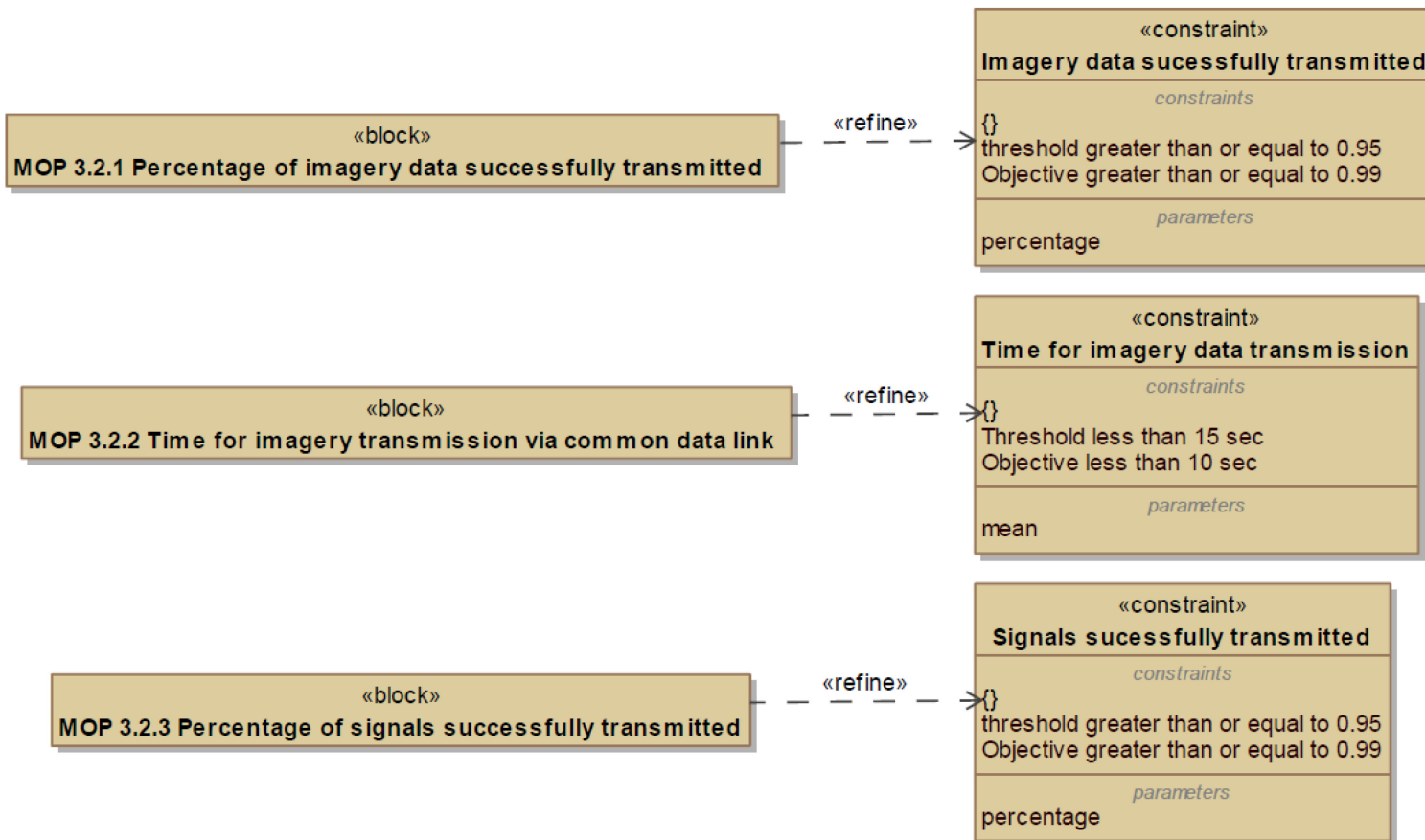


bdd [Block] COI 3 - Can the E-X Platform support exploitation and dissemination of ISR information to end users? [ COI 3 - Can the E-X Platform support exploitation and dissemination of ISR information to end users? ]

«block»  
MOE3.1 Generation Signal Products

«block»  
MOE3.2 Successful Transmission of Messages





bdd [Block] COI 4 - Is E-X Airborne Sensor Platform employable to support the TACS? [ COI 4 - Is E-X Airborne Sensor Platform employable to support the TACS? ]

«block»  
MOE 4.1 Generate Mission Plan

«block»  
MOE 4.2 Conduct Post Mission Analysis

«block»  
MOE 4.3 Conduct Training Analysis

bdd [Block] MOE 4.1 Generate Mission Plan [ MOE 4.1 ]

«block»  
MOS 4.1.1 Operator Rating of Mission Planning

«refine»

«constraint»  
Operator Assessment of Mission Planning

*constraints*

{  
Sufficient for the operator to conduct mission planning processing to support mission execution.

«block»  
MOS 4.1.2 Technician Rating of Mission Planning

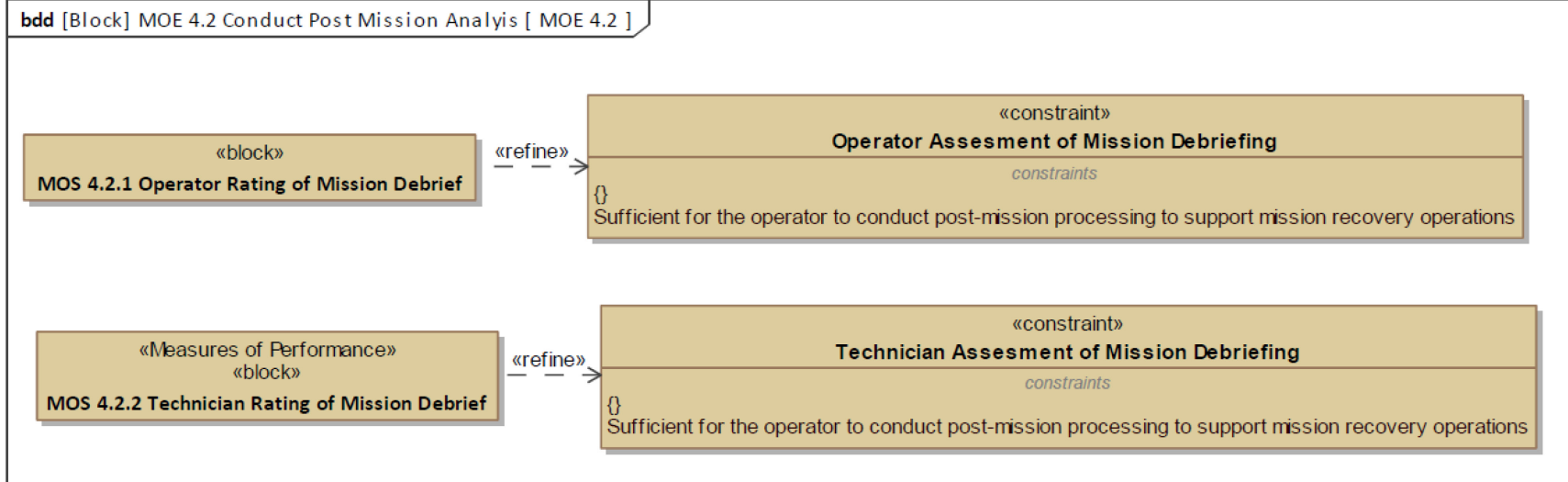
«refine»

«constraint»  
Technician Assessment of Mission Planning

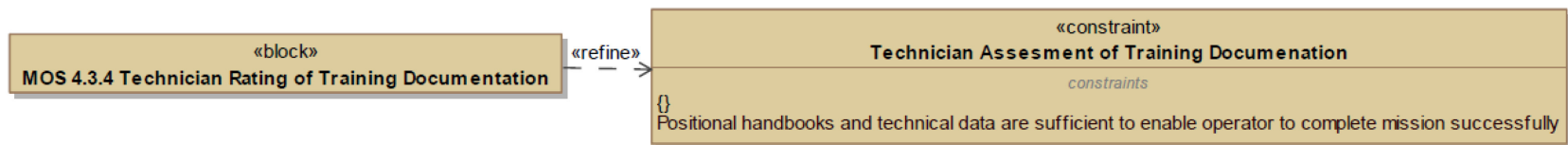
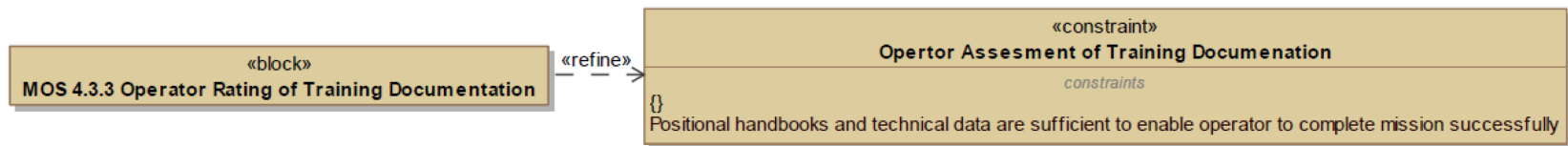
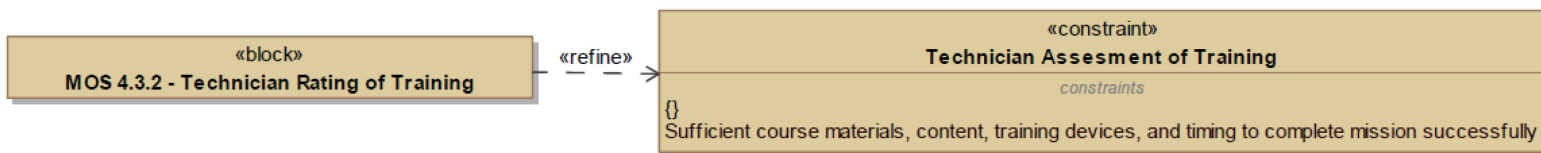
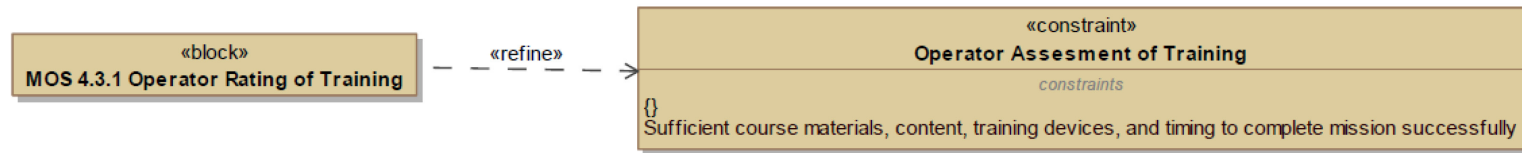
*constraints*

{  
Sufficient for the operator to conduct mission planning processing to support mission execution.





bdd [Block] MOE 4.3 Conduct Training Analysis [ MOE 4.3 ]



bdd [Block] COI 5 - Is E-X Airborne Sensor Platform sustainable to support the TACS? [ COI 5 - Is E-X Airborne Sensor Platform sustainable to support the TACS? ]

«block»  
**MOE 5.1 Aircraft generation and recovery**

