

REPORT DOCUMENTATION PAGE					Form Approved OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.						
1. REPORT DATE (DD-MM-YYYY) 08/29/2023		2. REPORT TYPE Technical Report			3. DATES COVERED (From - To)	
4. TITLE AND SUBTITLE Systems Engineering Processes to Test AI Right (SEPTAR) Release 1				5a. CONTRACT NUMBER W56KGU-18-D-0004		
				5b. GRANT NUMBER		
				5c. PROGRAM ELEMENT NUMBER		
6. AUTHOR(S) Carlos Balhana; Ivy Chen; Dr. Ronald W. Ferguson; Jim Lockett; Danny Moore; Carol Pomales; Flo Reeder				5d. PROJECT NUMBER		
				5e. TASK NUMBER W56KGU22F0017		
				5f. WORK UNIT NUMBER		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) The MITRE Corporation 7515 Colshire Drive McLean, VA 22102				8. PERFORMING ORGANIZATION REPORT NUMBER PRS-23-2503		
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) United States Army (ACC-APG)				10. SPONSOR/MONITOR'S ACRONYM(S) ACC-APG		
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)		
12. DISTRIBUTION/AVAILABILITY STATEMENT DISTRIBUTION STATEMENT A. Approved for public release: distribution unlimited.						
13. SUPPLEMENTARY NOTES Preprint submission. DOT&E in Partnership with CDAO, R&E CT, DTE&A, and TRMC Workshop: T&E of AI-A Systems						
14. ABSTRACT Systems Engineering Processes to Test AI Right (SEPTAR) presents benefits and best practices for proactive planning for Test and Evaluation (T&E) activities for AI-enabled system (AIES). By following these best practices, AIESs are more likely to be delivered on time, to meet budgetary goals, and to perform effectively to meet mission expectations.						
15. SUBJECT TERMS Systems Engineering Process; Artificial Intelligence; Systems Engineering Processes to Test AI Right (SEPTAR); AI-enabled system (AIES); best practices; test and evaluation (T&E); acquisition strategy; use case; requirements						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES 116	19a. NAME OF RESPONSIBLE PERSON Susan Carpenito	
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER (Include area code) 781-271-7646	



MTR230044 MITRE TECHNICAL REPORT

Systems Engineering Processes to Test AI Right (SEPTAR) Release 1

Sponsor: OUSD DTE&A
Project No.: 101074.23.401.D320.P04

The views, opinions and/or findings contained in this report are those of The MITRE Corporation and should not be construed as an official government position, policy, or decision, unless designated by other documentation.

This document was approved for public release, case number 23-2503. Distribution unlimited

©2023 The MITRE Corporation.
All rights reserved.

McLean, VA

Authors:

**Carlos Balhana
Ivy Chen
Dr. Ronald W. Ferguson
Jim Lockett
Dr. Danny Moore
Carol Pomaes
Dr. Flo Reeder**

August 2023

1	Executive Summary.....	1-1
2	Summary of Recommendations	2-2
2.1	Requirements for AIES.....	2-2
2.2	AIES Acquisition Strategy.....	2-2
2.3	AIES Development	2-2
2.4	Testing of AIES	2-2
2.5	Deployment/Sustainment of AIES.....	2-2
3	Introduction	3-3
3.1	DoD Acquisition Pathways and T&E of AIES.....	3-4
3.2	Document Organization	3-5
3.3	Scope & Assumptions.....	3-5
4	Unique Challenges of AI T&E.....	4-1
4.1	AI Defined	4-1
4.2	Differences in AIES T&E Tasks from Conventional Software.....	4-2
4.3	What Is Done Differently: The Test Mindset in AI.....	4-3
4.4	Difficulties in Assessing AIES Quality	4-4
4.4.1	Overall Measures of AI Enablement.....	4-4
4.5	Machine Learning Operations (MLOps) Development.....	4-5
5	Software Planning Phase	5-1
5.1	Requirements	5-2
5.1.1	Agile Requirements Methodology	5-4
5.1.2	Prototyping.....	5-6
5.1.3	AIES Hardware Infrastructure Requirements	5-6
5.2	Acquisition Strategy for AIES	5-7
5.2.1	Data Rights.....	5-8
5.2.2	Data Quality	5-9
5.2.3	Model Rights.....	5-10
5.2.4	Protections for AI Model Training Data	5-10
5.2.5	Communicating the Relevance of the AI Threat During Contracting	5-11
5.2.6	Sufficient Resources to Execute AI Development Program.....	5-11
5.2.7	Requests for Proposals (RFPs).....	5-13
5.3	Test Strategy	5-13
5.3.1	Test & Evaluation Strategy/TEMP	5-15
5.3.2	Test Planning with T&E Workshop.....	5-16

6	Software Execution Phase.....	6-1
6.1	Development and Model Testing.....	6-1
6.1.1	AIES “Cards”.....	6-1
6.1.2	AI Model testing during the Development Phase	6-4
6.2	T&E of AIES	6-18
6.2.1	Model Test & Evaluation.....	6-18
6.2.2	Cybersecurity Testing for AIES.....	6-19
6.2.3	Operational Test & Evaluation	6-21
6.3	Deployment, Sustainment and Performance Monitoring.....	6-22
6.3.1	Assessment and Authorization to enable the Authorization to Operate	6-22
6.3.2	AIES Fielding and Integration	6-23
6.3.3	Continuous T&E, Model Monitoring and Maintenance	6-25
7	Exemplar SEPTAR Use Cases	7-1
7.1	Perception (Use Case 1).....	7-1
7.1.1	Use Case Description for Image Recognition Task.....	7-2
7.1.2	Related T&E Impacts for Each Phase of the SELC for Use Case 1	7-3
7.2	Sensemaking (Use Case 2).....	7-8
7.2.1	Use Case Description for Document Triage using Named Entity Recognition (NER).....	7-8
7.2.2	Related T&E Impacts for Each Phase of the SELC for Use Case 2	7-9
7.3	Course of Action (COA) Generation (Use Case 3)	7-14
7.3.1	Use Case Description for COA Recommendation for Mission Planning.....	7-14
7.3.2	Related T&E Impacts for Each Phase of the SELC for Use Case 3	7-15
7.4	Summary of Use Case Analysis.....	7-20
Appendix A	Adaptive Acquisition Framework from DoD 5000.89	A-1
Appendix B	Key Differences between Traditional Software and AI Identified in GAO-22-104765	B-1
Appendix C	Ten Themes for a T&E Methodology Software and AI	C-1
Appendix D	Evaluation Card Template	D-1
Appendix E	Model Card Template.....	E-1
Appendix F	Data Card Template.....	F-1
Appendix G	Trustworthiness Metrics.....	G-1
Appendix H	Data Split Methods.....	H-2
Appendix I	Taxonomy for AI Cybersecurity and Common Threats to ML.....	I-1
Appendix J	Roadmap and Checklists	J-1

Appendix K	MLTS and Explanation Satisfaction Scale	K-1
Appendix L	CDAO T&E Workshop - Group Exercise	L-1
Appendix M	Acronyms and Abbreviations.....	M-1

List of Figures

Figure 1-1. Considerations across the SELC to Enable the T&E of AIES [8]	3-5
Figure 2-1. AI Categories of Application Area (Left) and AI Approaches & Algorithms (Right) [9]	4-1
Figure 2-2. The MLOps Cycle [16]	4-6
Figure 3-1. The Role of a Product Manager During Agile Requirements Development	5-5
Figure 3-2. Key Stakeholders Involved When Defining Requirements	5-5
Figure 3-3. AI/ML T&E Framework	5-14
Figure 6-1. Notional Card Linkages	6-2
Figure 6-2. Primary Elements for the AI T&E Evaluation Card	6-3
Figure 6-3. Data Splitting Schema.....	6-9
Figure 6-4. Algorithm for 10-Fold Cross Validation.....	6-11
Figure 6-5. Synthetic Data Approaches to Achieve Repeatability	6-12
Figure 7-1. ML Use Cases	7-1

List of Tables

Table 3-1. Roles across the SELC to Enable the T&E of AIES	5-1
Table 3-2. DoD and Contractor Roles for AIES.....	5-2
Table 6-1. Common Dataset Properties and Data Profiling Questions	6-6
Table 6-2. Confusion Matrix.....	6-15
Table 6-3. Performance Measures for Classifier Models	6-15
Table 6-4. Performance Measures for Regression Models	6-16
Table 6-5. Sustainment Areas and Continuous SELC Support Tasks	6-26
Table 7-1. Image Recognition Use Case Specifications	7-2
Table 7-2. Use Case 1 – Perception Summary Table	7-7
Table 7-3. Document Triage Use Case Specifications	7-8
Table 7-4. Use Case 2 – Sensemaking Summary Table	7-13
Table 7-5. COA Recommendation Use Case Specifications.....	7-15
Table 7-6. Use Case 3 – Course of Action Summary Table.....	7-19
Table 7-7. Use Case Differentiation	7-20
Table B-1. Key Differences between Traditional Software and Artificial Intelligence	B-1
Table C-1. Ten Themes for Test and Evaluation Methodology Software and AI.....	C-1
Table D-1. Sample Evaluation Card for Use Case 2	D-2
Table E-1. Model Card Example for Use Case 2.....	E-3
Table F-1. Data Card Example for Use Case 1.....	F-4
Table J-1. Sample AI T&E Roadmap	J-1
Table K-1. Explanation Satisfaction Scale [65].....	K-2

1 Executive Summary

The Department of Defense (DoD) is making sizable investments (\$14.7 billion) in Artificial Intelligence (AI) Research and Development (R&D) and acquiring AI through programs . Ensuring proper process execution enables these investments to be realized, especially the processes that ensure effective evaluation of the intended AI-enabled system (AIES). SEPTAR (Systems Engineering Processes to Test AI Right) presents benefits and best practices for proactive planning for Test and Evaluation (T&E) activities for AIES. By following these best practices, AIESs are more likely to be delivered on time, to meet budgetary goals, and to perform effectively to meet mission expectations.

Three major themes are targeted:

1. Broadening the T&E continuum
2. Defining data needs for AIES up front
3. Evaluating the Systems Engineering Life Cycle (SELC) to inform AIES trustworthiness

Assuring and understanding processes used earlier on in the systems engineering lifecycle to build the AIES informs T&E later during testing and development. AIES need continuous monitoring and analysis to ensure mission needs are being met throughout the life of the system and especially during deployment and sustainment.

SEPTAR is divided into the five phases of the SELC and specific recommendations are provided for each phase to enable T&E:

1. Requirements for AIES
2. AIES Acquisition Strategy
3. AIES Development
4. Testing of AIES
5. Deployment/Sustainment of AIES

SEPTAR also presents specific use cases and how SEPTAR best practices can be used to improve T&E for an AIES.

SEPTAR is a living document by its very nature and cannot be comprehensive of all relevant concerns. The recommendations and best practices contained in this report serve as a foundation for the DOD testing community to become better equipped to adapt to the rapidly increasing adoption of AIES.

2 Summary of Recommendations

This section provides a high-level overview of the T&E recommendations contained in this report. These best practices have been developed specifically for implementing AIES. This list is not comprehensive, and the full report provides a more in-depth discussion including rationale for each these recommendations. Refer to Sections 5 and 6 which discusses recommendations specific to each area of the SELC.

2.1 Requirements for AIES

- Consult users and domain experts for system design insights
- Explore prototypes and data features
- Consider hardware performance limitations
- Focus on mission-relevant use cases
- Assess data availability

2.2 AIES Acquisition Strategy

- Establish data and model rights, ensure access to evaluation, data and model cards
- Define expectations using evaluation, model, and data cards
- Advise on assessment methods, such as adversarial testing, for AI systems and models
- Be aware of cybersecurity risks related to AI
- Define resources and roles, including model sustainment

2.3 AIES Development

- Test the AI model within the AI-enabled system and share data with DOD T&E
- Enable infrastructure integration (MLOps to DevSecOps)
- Prepare data with proper labeling and ensure operational representation, separate training and testing data
- Implement iterative model training and testing

2.4 Testing of AIES

- Review the AI model's train/test data composition during DT&E
- Test the AI model integrated into the AI-enabled system
- Gather user feedback to build trust
- Evaluate MLOps pipeline and practices as part of cybersecurity
- Ensure validation of safe performance

2.5 Deployment/Sustainment of AIES

- Assess and obtain ATO including unique considerations of AI components of the system
- Ensure users are properly trained
- Collect logs and training data
- Implement model drift detection and establish performance guardrails

3 Introduction

This report was developed by a MITRE team supporting Developmental Test Evaluation and Analysis (DTE&A). The MITRE team conducted extensive reviews with other AIES subject matter experts (SMEs) across DOD organizations (DOT&E, CDAO, DTE&E), academic partners including Virginia Polytechnic Institute and State University, Georgia Tech Research Institute, Carnegie Mellon University, and Johns Hopkins University Applied Physics Laboratory and other Federally Funded Research and Development Center operators (Aerospace Corporation, Institute for Defense Analyses), and the Scientific Test and Analysis Techniques Center of Excellence.

The report presents the recommended approach that programs will follow to best inform and enable Test and Evaluation (T&E) of AIESs. By following the recommendations set forth in this report, we expect that AIESs are more likely to be delivered on time, to meet budgetary goals, and to perform effectively to meet mission expectations. We have focused mostly upon the systems engineering lifecycle (SELC) for Machine Learning (ML) forms of AI, but many recommendations can be considered relevant for other forms of AI as well. (Refer to section 4.1 for an explanation of the taxonomy we use for AI, including ML.)

Three major themes are explored in this report include:

1. Broadening the T&E continuum
2. Defining data needs for AIES up front
3. Evaluating the SELC to inform AIES trustworthiness

The document is primarily developed for test engineers and personnel involved in the early stages of SDLC to strategize, plan, and execute test and evaluation activities for AIES. From the organization of this report, program/project managers will recognize unique considerations for AIES requirements, acquisition needs, and post-deployment maintenance. Development teams can also better understand the key impacts and leading preparations required to jointly achieve successful T&E for AIES.

The differences between systems engineering process for an AIES versus standard best practices for systems engineering are blurred. Wherever possible, we have tried to emphasize best practices that are unique for an AIES, but in some cases, best practices for an AIES may serve as standard best practices for SELC delivery. Throughout the report, and especially during the requirements phase, we emphasize AI is an enabling capability, not a product/goal, and it is important to recognize that sometimes AI capabilities may not be the best solution(s) to achieve mission objectives.

While this report primarily focuses on best practices within the software acquisition pathway, most of the recommendations can be applied to other acquisition pathways as well, and are not limited to just one type. Additionally, this report should be considered as a starting point that can be adapted to accommodate different operational implementations of AIES (e.g., centralized systems requiring more compute power or storage, and other AIES which may be more distributed). Although each implementation may have specific T&E requirements to account for usage, these can be addressed in use case requirements.

Broadening the T&E continuum

Broadening the T&E continuum for AIES is part of DTE&A. T&E is now considered a

continuum rather than a challenging phase at the tail end of a program lifecycle. When this approach is applied to AIES, T&E concepts begin to occur at the start of a system's lifecycle, allowing plans for access to the information needed to most effectively assess the AIES. This also includes expanding the scope of technical and managerial activities to include T&E relevant considerations, such as model training/test data management. Subject matter experts from both academia and commercial industry promote the paradigm of both “shifting left *and* right” for effective performance implementation and management of an AIES. Shifting right refers to the need to monitor, iterate, and make changes to the AIES even once deployed into operations.

Defining data needs for AIES up front

Establishing data needs early in the AIES project lifecycle is essential to ensuring successful implementation of AIES programs. The lack of a SELC-wide data strategy in place is the chief reason most AI projects fail. T&E personnel can avert potential adverse SELC outcomes by assessing whether current data practices contribute positively on system development, model performance, and operational requirements. Continuous T&E tasking further ensures that observable metrics and data requirements are closely aligned with key performance parameters (KPPs). Such assessments facilitate the identification of faulty or ill-suited data practices and enable greater flexibility within an agile framework.

Evaluating the SELC to inform AIES trustworthiness

Programs that are delivering AIES should take key steps to ensure a successful execution from start to operationalization through on-going AIES monitoring and sustainment. Assessing the processes followed to design, train, and integrate the AIES can be used to inform the trustworthiness of the overall AIES.

3.1 DoD Acquisition Pathways and T&E of AIES

The DoD acquisition process has evolved (see Appendix A) to include several more agile approaches including the Software Acquisition Pathway (DOD Instruction 5000.87). The Software Acquisition Pathway (SAP) is well-suited for AIES due to its rapid and iterative approach to delivering capabilities. This pathway can accommodate the evolving nature of AIES' effectiveness and mission. AI model performance, the AIES system, training data, and the ML development process should all be included as part of the shared body of evidence for AIES. Additional recommendations for changes are discussed throughout this document, framed within this Acquisition path. The following figure summarizes the recommendations to enable the T&E of AIES across the SAP.

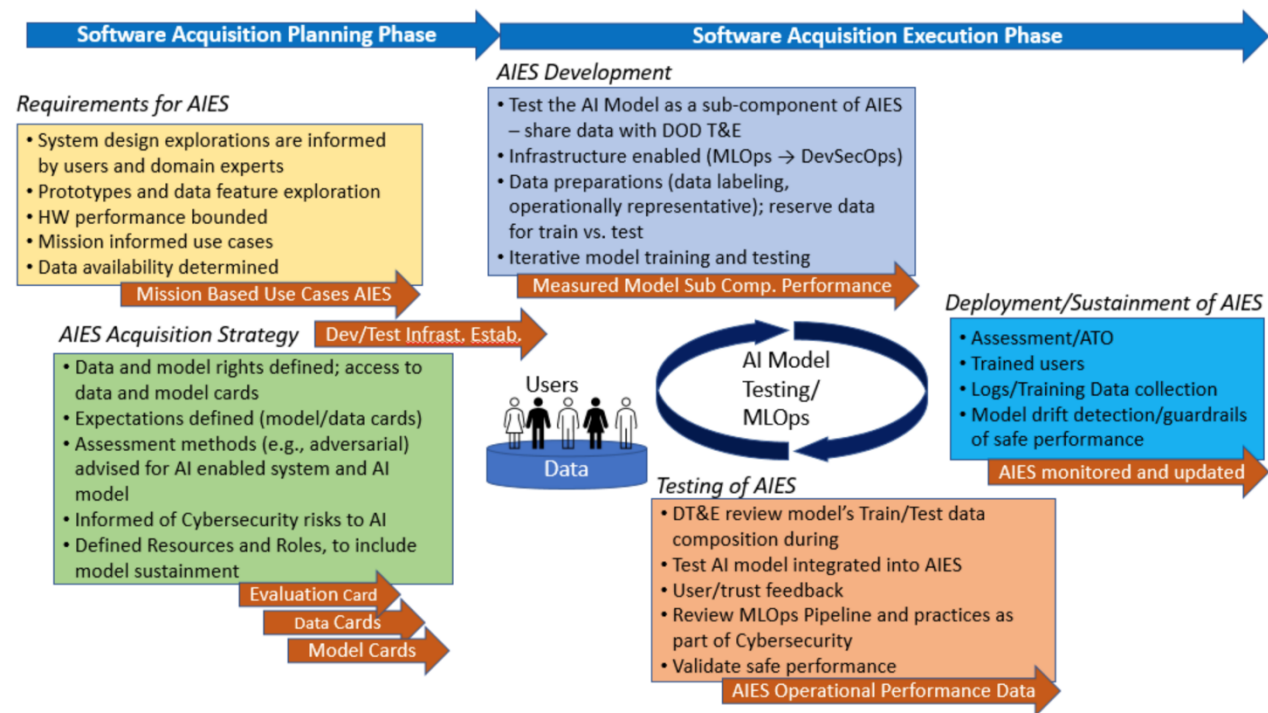


Figure 3-1. Considerations across the SELC to Enable the T&E of AIES

3.2 Document Organization

The document is organized into the following sections:

Section 4 explores the distinctions in AI and with T&E. AI is defined while comparing the features between AI and traditional T&E activity, including common barriers and challenges.

Section 5 describes the Software Acquisition Pathway Planning Phase. This includes Requirements for AIES and Acquisition Strategy.

Section 6 describes the Software Acquisition Pathway Execution Phase. This includes AIES Development, Testing of the AIES, and Deployment/Sustainment of the AIES.

Section 7 provides three selected use cases to illustrate the benefit of the SELC for AI and T&E. The first is a perception use case, the second is a sensemaking use case, and the third is a course of action generation use case. Each use case will demonstrate the advantages to thinking in terms of a lifecycle with necessary T&E considerations.

Following Section 7, there is supporting content with a summary, bibliography, and supplementary appendices.

3.3 Scope & Assumptions

The domain of T&E for AIES and its impact on the SELC is evolving rapidly, and there are many divergent positions on what best practices are for T&E resources. The report includes relevant information that is mature enough to be included at this time and will require updates to address emerging information in this rapidly changing area.

The report includes limited discussion on autonomy found in robotic systems or other safety-critical systems and is primarily focused on AIES that serve the purpose of making and expressing inferences based on input data. Operational use cases are discussed throughout the report. Section 7 includes three exemplary use cases that were chosen to reflect data reasoning scenarios not specifically tied to autonomy, although they may fit into a larger workflow.

AI capabilities can be pursued through local SMEs augmenting their work directly with more discrete and stand-alone AI pre-acquisition prototype applications. AI can be pursued through the application of AI as part of a larger software system where the AI is less transparent to the user and tester. This report largely focuses on the latter and the changes to the overall process to ensure DoD test evidence can be met to characterize the overall system, including the AI.

Throughout the report, we have used the term “Data Card,” although we recognize that the term Data Sheet is often used in academic research and various applications of AI. We chose to use the term Data Card because we want to emphasize the importance of Evaluation Cards and the relationship between Evaluation Cards, Data Cards, and Model Cards and the need for T&E personnel to coordinate these critical components of the T&E process.

The paper "A Sensible Regulatory Framework for AI Security" by MITRE provides a valuable starting point for discussions about how to regulate AI in a way that protects the public good. SEPTAR complements this regulatory perspective by presenting an approach for developing and implementing trustworthy AI systems through evaluation, which ensures the technical robustness and ethical alignment of these systems with ethical principles.

4 Unique Challenges of AI T&E

4.1 AI Defined

The Institute of Electrical and Electronics Engineers describes AI as “a multidisciplinary field whose goal is to automate activities that presently require human intelligence” [2]. In subsequent sections we will describe AI tasks as well as the differences between AI and more deterministic software that increases the T&E complexity.

Experts have organized the AI field by the application area or by the type of AI algorithm or approach (Figure 4-1.). Application area categorization considers the nature of the task and type of data involved. For example, computer vision involves mostly visual classification and reasoning tasks over image and video data, while natural language processing involves manipulating textual data. Similarly, AI approach categorization considers AI algorithm type, including whether it involves an explicit representation of the task and reasoning (i.e., symbolic AI) or performs statistical learning over data (i.e., ML). Recent developments in Generative AI have expanded the ML categories, as generative approaches involve a combination of self-learning from supervised, unsupervised, and reinforcement learning methods to create systems that can generate new data and learn from that data. ML methods and knowledge representation and reasoning methods can be applied across multiple application areas. Refer to for an expanded taxonomy.

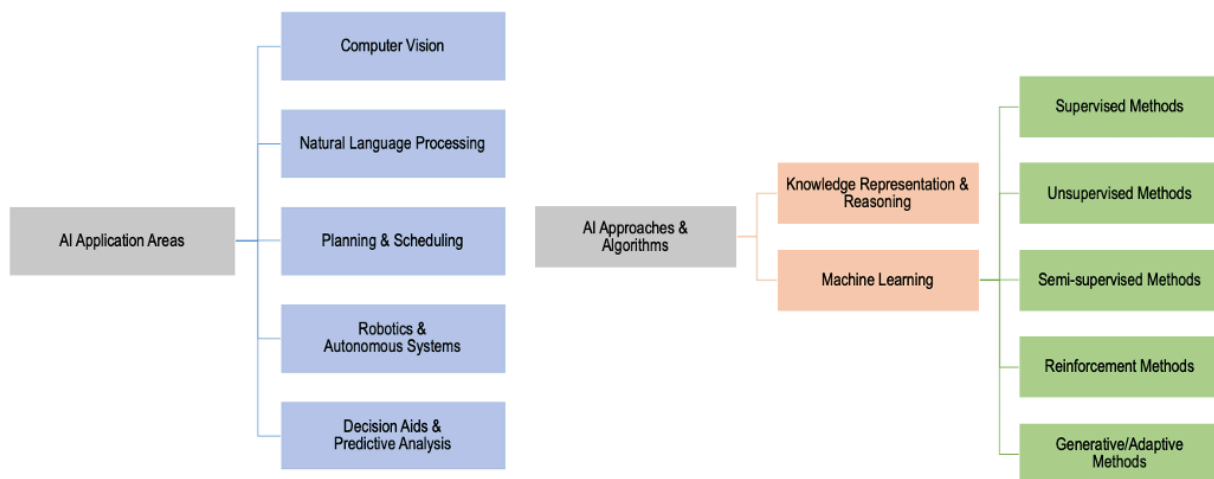


Figure 4-1. AI Categories of Application Area (Left) and AI Approaches & Algorithms (Right)

Relative to particular cognitive tasks, AI applications can be categorized as perception, sensemaking, and course of action generation. The use cases in this document represents each of these categories through the lens of a specific application type (Section 7).

These categories provide some broad guidance on testing metrics. Each AI application area has standard testing metrics which can be reused for similar applications, independent of approach. Similarly, AI approaches may have standard testing metrics that can be reused independent of application area. For human-AI teaming systems, cognitive metrics may be useful for judging the human-AI team, such as cognitive load or user trust. Successful T&E regimens likely combine all three metric types, as it is often more meaningful and informative than just any single metric and allows for any T&E unique to the project.

4.2 Differences in AIES T&E Tasks from Conventional Software

T&E for AIES differs from that for traditional software. Traditional software is deterministic, meaning that given any set of inputs, the path of the software is determined by the code as written. Identical inputs will always give identical outputs. This determinism allows T&E with relatively limited test sets, since a test case that was passed previously should always work correctly if code is unchanged. The statistical nature of AI systems (particularly, ML) means that system correctness can only be estimated relative to the available test data. ML systems might give different outputs for identical or very similar inputs. Therefore, a large test corpus may be needed to estimate the level of correctness for the model.

ML models such as neural networks, are relatively opaque, since at their foundation they rely on layers of hidden elements within the network that may not correspond directly to simple concepts or features. This makes it difficult to isolate error causes. By contrast, traditional software systems have a clear logical flow that can be analyzed to track down incorrect results.

ML development includes data analysis and curation in addition to the software that runs the model. In conventional software, the code is the basic item for analysis. ML, on the other hand, relies on models built from data in an iterative process. For each iteration, the model performance is measured on the “testing” data set with a separate hold-out data set for final evaluation once the model is deemed to perform at the desired levels. The data itself must be analyzed along dimensions such as category membership, edge cases, and completeness. This means that ML T&E involves closer inspection of the development process, including training code and data. Since training represents a significant resource cost, the ability to measure these becomes part of the T&E process. For a typical ML workflow development, training produces an evaluation report that would additionally include:

- Performance of established metric on a validation dataset
- Plots, such as precision-recall curves
- Operational statistics, such as inference speed
- Examples where the model was most confidently incorrect
- Deterministic software testing, particularly on input conditioning and output presentation

With conventions such as:

- Documenting all hyperparameters used to train the model
- Promoting only models that offer an improvement over the baseline model
- Models are trained, validated, and tested iteratively.
- Performance monitoring should occur throughout the process.
- The desired outcome from ML testing is for all datasets to have similar representative properties of the target domain.

The challenges and differences in the distinctions between conventional software versus AIES testing include:

- AIES are inherently complex where possible outcomes are vast. Some AI methodologies are probabilistic or have a level of uncertainty associated with results.
- Conventional software is expected to produce consistent and expected answers for every possible input. For AIES, the requirement is to produce answers that are close enough, often enough, over a wide enough range of inputs where “enough” can vary.
- Variation in AIES outputs can result from both changes in training and testing data and code whereas in conventional software the output variations result only from code changes.
- Data can have bias, which can become embedded in several ways within an AIES.
- Traditional software quality models do not align neatly to AIES. While an AIES may provide a correct answer, it may give similar or different results from the same inputs.

We must re-think test coverage to revise our aperture for completeness of testing over the continuum of the lifecycle. The U.S. Government Accountability Office (GAO) recently contrasted AI software development from traditional development in a recent report on AI for weapons development :

- AI requires vast amounts of data for development and continuous monitoring after deployment.
- AI development must be both rapid and continuous to keep pace with changing technologies.
- AI capabilities are subject to difficulties in the transition from R&D to end user.
- It is necessary to manage the full lifecycle across a system to be effective and operational.
- AI requires maintenance to ensure the sustainment phase of a project, as models may drift, requiring updates.
- The data challenges and model maintenance problems make it essential for data analysis and insight creation as the first step toward data-driven decision making.

4.3 What Is Done Differently: The Test Mindset in AI

The development of AIES requires a different approach from conventional software testing that further emphasizes T&E because of the experiment-oriented nature of ML. Conventional software testing applies, since AIES contain software for a variety of functions such as data preprocessing, result communication, and user interface. Since AIES, particularly ML-based systems, are sensitive to the training data, the need to have more complex and continuous testing is important. The DoD “Responsible AI Strategy and Implementation Pathway” specifically states “Develop a Testing, Evaluation, Validation & Verification (TEVV) framework to articulate how test and evaluation (T&E) should be intertwined across an AI capability's lifecycle and pathways for continuous testing and standards for DOT&E documentation and reporting” . AIES testing needs to be continuous and integrated throughout the lifecycle of a system.

Since AIES testing is more complex, the required testing documentation is more extensive. It includes Evaluation Cards, which capture the overall AI testing specifics; Data Cards that specify both training and evaluation data; and Model Cards that describe the training process for models. These will be described in detail in later sections. Since T&E occurs throughout the systems engineering process, Evaluation Cards can be populated throughout the process and T&E may need more than one Evaluation Card.

4.4 Difficulties in Assessing AIES Quality

While traditional software engineering practices mandate the formalization of technical requirements before development efforts may begin, initial AIES development can proceed before system specifications are fully or clearly defined. Consequently, T&E teams may not be able to assess system quality properly without objectively established system requirements. Such cases of premature development may result from an ill-informed understanding of AIES capabilities, data complexity, human workflow, or a combination of these. We may broadly group these challenges into two distinct categories: task-dependent challenges and user-dependent challenges.

AIES tasks may lack a determinable set of valid reference values. For example, in human language translation, a variety of translations of a text may be valid. Machine translation system output, therefore, cannot be measured against a single correct answer since no one single correct answer exists. Some AIES use cases, including course-of-action recommender systems, will frequently fuse features of distinct data types such as text, audio, video, and assorted sensor-based signal measurements which increase overall system complexity. Such systems may therefore require several sub-evaluations to properly assess how system quality may be impacted by a specific subset of inputs. AIES T&E efforts are also further complicated by the inherent opacity of most neural network, or deep learning, architectures. These models are comprised of multiple layers and hidden states which cannot be directly inspected, making it difficult to assess whether a trained model is sensitive to the same set of conditions and criteria that human experts would factor in their own predictions and decision-making processes.

Several challenges emanate from AIES user communities. AIES capabilities provide added value only insofar as they can reasonably automate human activities for mission-driven objectives, which can be difficult to quantify. Data inputs may change over time, feeding into a model trained under significantly different conditions; such divergences often lead to what is called conceptual drift. Nearly all AIES outputs must undergo some level of human-in-the-loop review to validate model-generated predictions and prevent potentially undesirable outcomes, but many ML models do not provide adequate transparency into output generation. Such inscrutability can often inadvertently lead to pro-automation bias whereby end users place unwarranted trust in system outputs. Lastly, data collection and curation practices may reflect unrecognized human biases, which may impact system quality either from the outset or over time. Such biases may not be captured or accounted for during system testing and thereby introduce additional confounding factors during the evaluation process.

4.4.1 Overall Measures of AI Enablement

Along with assessing the system itself, it is recommended that an assessment of the build process also be performed. This trustworthiness assessment measures confidence in the development

process, and thus also informs the system's risk (see Appendix G. Trustworthiness Metrics). The checklist in the trustworthiness assessment can be used during developmental or operational test activities to inform the independent tester to scope their assessment of the AIES.

In addition, the overall process should be assessed **qualitatively** to assess the trustworthiness of the AIES. Following are activities to look for to help assess this:

- *Requirements*: Inclusion of users and domain experts involved in model feature selection, system exposure of data to enable trust, hardware performance bounds (central processing unit [CPU]/graphical processing unit [GPU]), production debug enablement, optimal model type to be selected for the problem that has occurred, mission informed use cases, and system self-restraint through appropriate guardrails.
- *Contracting for AI*: Contract language that defines data rights, model rights, DoD/contractor roles for sustaining the AI-enabled system, protection of data, advising of expected assessment of the system to include data beyond expected bounds of performance and adversarial data, and any implication to the contractor/DOD should the system perform harm.
- *Data*: Data cleaned, labeled, and reviewed for bias/quality; data schema defined and applied to new training data.
- *Model Training and Evaluation*: ML pipeline review and auditing, logging of training; training with adversarial data, model drift, or degraded performance detection.
- *Model Deployment*: Logs, model drift detection, user feedback/trust feedback, model applicability to mission.
- *Security*: Review and assessment of evidence of data and model risks and further protection processes. Level of design that inherently handles the known threats and handles or alerts the user to these risks.

The assessment requirements would be shared with the vendor and sought (along with evidence) to support their responses. These results will help to inform on risks and define mitigations that may be warranted, depending upon the mission and application of the AIES.

4.5 Machine Learning Operations (MLOps) Development

In Sections 6.2 through 6.4, we have outlined several key differences that distinguish AIES solutions from conventional development for T&E scenarios. The MLOps framework provides a system-oriented approach specifically tailored to AIES, that may facilitate more effective T&E. It specifies a cyclical, iterative path for developing, evaluating, integrating, and monitoring models throughout the operational course of a system's lifecycle. Given that data domains, analytical use cases, and mission-driven priorities can change over time, the MLOps framework prescribes a set of practices to facilitate sustainable AIES maintenance long after AIES solutions are initially deployed. Figure 4-2 depicts a broad representation of the key processes of the MLOps cycle, grouped into four main phases: data profiling, training and tuning, continuous assessment, and feedback. The MLOps cycle presupposes that all four phases are to be repeatedly performed on a periodic basis to ensure sustainable long-term AIES optimization.

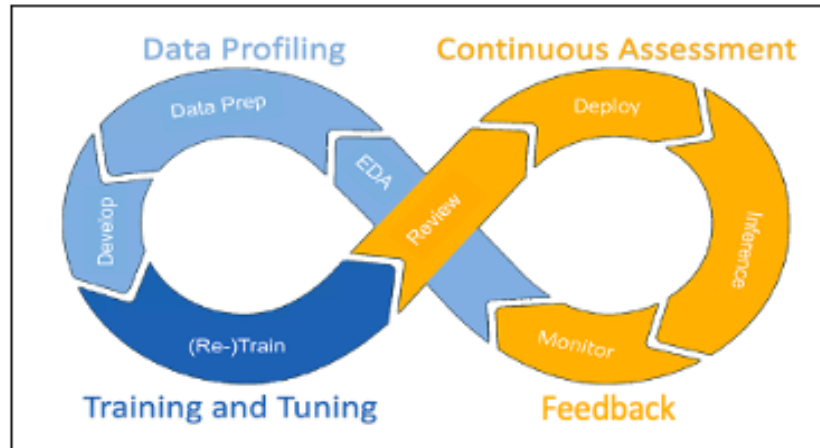


Figure 4-2. The MLOps Cycle

Crucially, the MLOps framework requires T&E personnel to fulfill a pivotal role in the continuous assessment and feedback phases. Under an agile-compliant setting, T&E staff must work closely with data scientists, domain experts, system developers, system integrators, and end users to address emerging AIES support needs over the SELC.

Given the overarching scope of the MLOps cycle, detailed discussions of its role in T&E will occur in the following sections:

- Data profiling: Sections 6.2.2 and 6.2.3
- Training and Tuning: Section 6.2.4
- Continuous Assessment and Feedback: Section 6.3.3

5 Software Planning Phase

The following sections discuss considerations and activities that should occur at each stage of an AIES project to enable the test and evaluation of a system framed within the DoD Software Acquisition Pathway, as it is most aligned with the agile and iterative nature of ML-based AIES. Additional specific recommendations for improvements to the SELC changes, which are more narrowly applicable to even more specific types of ML applications (perception, sensemaking, course of action analysis generation), are shown in the Use Cases, Section 7.

The specific tasks and roles most uniquely needed to enable AIES are defined in Table 5-1. This table was developed by the authors and serves as an example; it is not prescriptive. Additional roles such as users, legal, adjacent organizations, higher level staff, and those affected by the AIES can be added. A best practice is to develop a Responsibility, Accountability, Consulted, Informed matrix as early as possible during the requirements phase prior to acquisition.

Table 5-1. Roles across the SELC to Enable the T&E of AIES

		DOD PMO							DOD: Independent Test			Contractor		
	Tasks	Contract Officer	Functional Sponsor	User	Program Manager	Data Engineer	Dev Ops Engineer	Cyber Security	Dev Test Engr	Cyber-Security	OTA	Lead Developer	Data Engr/ CTR	Lead Test
Requirements	AI Prototyping and Exploration	I	R	C	A	C	C	C	C	I	I			
	Requirements/ Use Case Development		R	C	A	C	C	C	C	C	C	I	I	I
Contracting	Ensures Contract Language includes AIES considerations	R	I	C	A	C	C	C	C	C	C			
	Collect, Assess, Label, Prep, Split, Curate Model Training Data	I	C	C	A	R*	C	C	C	C	I	I	R*	C
	Model Dev/Test Infrastructure	I	I	C	A	C	R*	C	C	I	I	R*	C	I
	Model Training/Test Infrastructure	I	I	C	A	R*	C	C	C	I	I	I	R*	I
Development	Evaluation Card	I	C	C	A	C		C	R	C	C	C	C	C
	Model Cards	I	I	I	A	C		C	C	C	I	C	R	C
	Data Cards	I	I	C	A	R*		C	C	C	C	C	R*	C
	AI Model Testing	I	C	C	A	C		C	R	C	C	C	C	C
Testing	AIES Testing (and Model Training Process Review)	I	I	C	A	C		C	R (DT&E)	R (CVPA/AA)	R (OT&E)	C	C	R (Unit)
Deployment and Sustainment	Deployed System, Model Monitoring	I	I	C	A	R*		C	I	I		I	R*	C
	Deployed System, Model retraining and redeployment	I	I	C	A	R*		C	C	C	I	C	R*	C

Responsible (R): Does the work to complete the task. **Accountable (A):** Delegates work and performs final review of completed deliverable. **Consulted (C):** Provides review and consultation. **Informed (I):** Kept informed of project process.

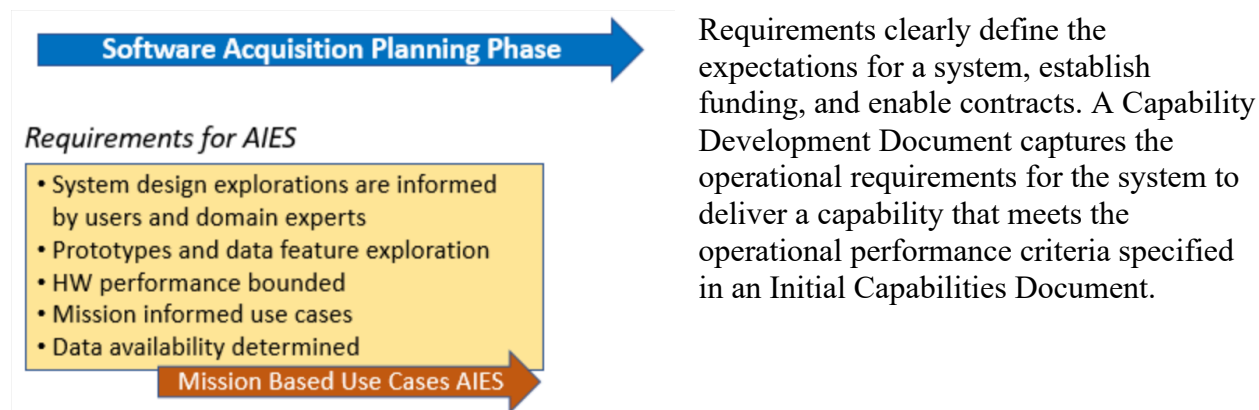
* Indicates a decision must be made if the activity is DoD or Contractor Led, and/or if a split of responsibilities exists.

To further amplify the unique considerations for AIES, Table 5-2. DoD and Contractor Roles for AIES calls out some of these roles and defines priorities and needs to enable them to be successful.

Table 5-2. DoD and Contractor Roles for AIES

DOD and Contractor Roles for AIES	
Data Engineer (broadly defined to include Data Scientist)	Builds scalable pipelines for model training and enables model integration. Lead to ensure data is generated and reflects the mission defined use cases. Detect bias in training data before affecting models, and to the extent possible, mitigate it. Role continues into deployment to monitor, acquire additional training data, retrain, and redeploy the model.
Cognitive Engineer	Performing cognitive analyses (such as workflow analysis) to help determine the effectiveness of potential AI solutions for human-AI teams. Evaluating performance of AI-human teams in cognitive terms, such as user trust and cognitive load.

5.1 Requirements



This section covers the following T&E-relevant topics:

Agile Requirements Methodology – T&E staff need to advise the requirements process to ensure that the operational environments where the AIES will perform are clearly captured in user stories.

Prototyping – Developing a prototype is recommended to validate assumptions and assist in determining if an AI-enabled solution is necessary and feasible. T&E personnel should start advising on the potential testing for specific types of AIES to enable informed selections for understanding T&E complexities and impacts to cost and schedule.

System Requirements – Defining the infrastructure and hardware needs, which typically include higher computing power and larger data storage capacities. Some systems may be resource constrained and these limitations should be defined to ensure effective design and T&E within these bounds.

Executive Order 13859, Feb 2019: *Maintaining Leadership in Artificial Intelligence* stipulates requirements for AI programs must exist for protecting the safety, security, privacy, and confidentiality of those involved or affected by the work accomplished.

The challenge with AIES is that component-level requirements may be unknown during the planning phase. They can change during ML model development. Furthermore, the application

of an AIES may change after the system has been fully implemented and is fully operational. Requirements can change rapidly for an AIES and as new uses for the system are identified. AIES are developed so that they evolve their behavior during the life of the system due to models that learn by themselves. Deep learning enabled systems can potentially change functional requirements while the system is operational. Consequently, continual evaluation against the requirements is essential. Therefore, T&E processes should occur throughout the lifecycle of a system until it is no longer operational. AIES need to be developed iteratively and generally rely on agile development best practices throughout the SDLC.

The International Data Corporation conducted a survey with companies implementing AIES and determined that most AI projects are not implemented in production because expectations are not well communicated due to a misalignment between technical and business teams. Establishing clear mission-driven requirements up front, where users, leadership, and technical teams build consensus reduces the risk of misaligned expectations and, therefore, increases the likelihood of successful implementations. AI should not be the goal; instead, the focus should be on delivering mission-driven use cases.

Requirements must begin with a robust problem statement that ensures the appropriate problem is being solved with the right solution. Many sources can help a team build problem statements. The Joint Capabilities Integration and Development System (JCIDS) is the DoD's most formal process for defining large acquisition requirements. The Software Acquisition Pathway provides an example of guiding the process of defining Capability Need Statements (CNS). This is a high-level capture of the need with enough information to define the solution as well as the threat environment. This can be drafted during the planning phase and approved before moving to the execution phase. There are also solicitation methods, such as Commercial Solution Opening (CSO), for smaller acquisitions. CSOs can help identify and acquire innovative solutions in which vendors may have ideas that potentially influence the problem statement. Broad Agency Announcements (BAAs) represent another approach for soliciting proposals for research and development, which can help the government acquire relevant capabilities and refine problem statements. Once a well-defined problem statement and set of requirements are established, the program office can proceed to the contracting phase.

A key resource for problem framing and/or requirement development is MITRE's Human Machine Teaming Systems Engineering Guide. This guide provides applied research findings for developers of automation, autonomy, and AI technologies, including a framework organizing human machine teaming research, methodology for engaging with users, and a set of adaptable requirements. Using knowledge elicitation techniques with experts and users can uncover aspects of human interaction with an AIES, which can further supplement other requirement development activities. Furthermore, Human Machine Teaming (HMT) can help accelerate up training of both the AIES and its user.

Human Machine Teaming can assist in ensuring trust is bidirectional. Users must trust an AIES to perform its functions correctly, safely, and reliably. Conversely, the AIES should also be designed to "trust" its users. This doesn't mean the system has emotions or beliefs; rather it means the system is designed to act upon user inputs and decisions under the presumption that they are valid and reliable. Requirements should be defined to validate both the system's functionality and its interaction with users.

Most AIES require continuous integration and continuous delivery (CI/CD), which involves continual reassessment and iterative updates throughout the SELC. Consequently, defining requirements prior to contracting and procurement is important to ensure AIES sustainment. This includes ensuring the identification of operations and maintenance (O&M) needs during the planning phase before contracting. Negotiating O&M, including T&E, post-deployment can be cost prohibitive.

Developing mission-informed use cases during the requirements phase informs all stakeholders in the software acquisition pathway. Use cases can align expectations between stakeholders and help inform a procurement team and potential future vendors of the objectives to be achieved. T&E staff should work with product managers to coordinate users, data scientists, developers, and leadership to create use cases. DoD has a responsibility to “mitigate potential risk during project development”. Well-defined use cases serve to mitigate potential risk.

It's important to include requirements for these operational use cases, including when, why, and how the AIES should turn full control back to the human, including how much context and lead time is needed for the human operator to come fully up to speed on a problem situation and make the best possible decision(s) to correct it. While the human is in control, the AIES can function like a pass-through, observing the human and recording their actions, until the human is satisfied that the problem is solved and hands control back over to the AIES. Involving the user in these use cases allows for a self-audit of the system. Incorporating best practices such as Human Machine Teaming can assist in this process.

5.1.1 Agile Requirements Methodology

Agile requirements methodology is the desired approach for identifying requirements for an AIES because developing ML models requires an iterative approach. Following an agile approach is relevant to AIES since it affords incremental software (SW) including model updates. Regardless of the specific agile methodology employed, the approach to team building should also embody agility. This involves including a diverse group of stakeholders at all stages of the Systems Engineering Life Cycle (SELC) and fostering teams that embrace and operate with an agile mindset.

A product manager's role is at the heart of agile methodology. A product manager is responsible for defining requirements for a project. Within the context of agile implementation, product managers identify system needs ("what" needs to be developed) and define success criteria through requirements. If a product manager is not identified, then the program manager is responsible for ensuring requirements meet mission needs.

While government projects typically have program managers, introducing product management best practices from industry can yield great benefits and reduce overall risk for AI-enabled projects. Both industry product and government program managers have the responsibility to synchronize expectations and needs among users, subject matter experts, data scientists, and systems engineers. By working closely with the program manager, T&E staff can ensure a system meets its objectives for all stakeholders, particularly with user representation, which is a critical factor in project success.

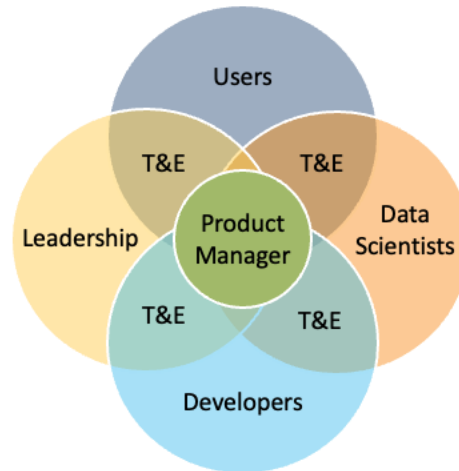


Figure 5-1. The Role of a Product Manager During Agile Requirements Development

Product managers must have a strong understanding of business needs and are critical for communicating these needs. A leading factor in AI project success or failure is the success of communicating expectations between leadership, users, and implementation teams. With the added AIES complexity, the role of a product manager is more critical to success. Only after a product manager has articulated requirements can a development team define requirements that are then designed, implemented, and tested. The inherently iterative agile requirements process makes it effective for AIES.

The AIES requirements definition process must include close collaboration between developers, T&E personnel, and operations staff in the build-integrate-test-deployment process. T&E personnel participation is necessary during requirements definition prior to contracting. For AIES, data scientists and data engineers are required during this phase (Figure 5-2. Key Stakeholders Involved When Defining Requirements).

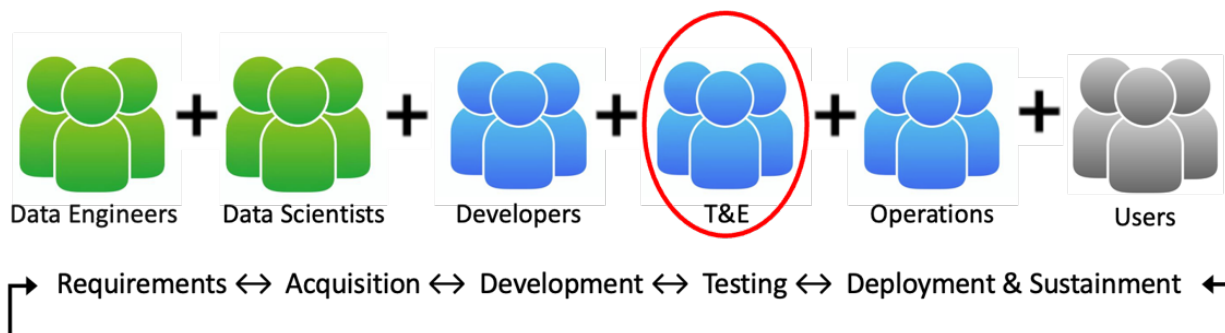


Figure 5-2. Key Stakeholders Involved When Defining Requirements

Collaboration between stakeholders during the requirements process is even more necessary when developing requirements for AIES. Successful AIES almost always require consultation with data scientists to determine whether an ML algorithm is achievable based on the availability of data etc. Product/program managers should be cautious not to assume that AI functionality is needed and should rely on their data scientists and developers to make this determination. The role of product manager is to define the business requirements and not the technical solution. The blurring of the role of a product manager with a developer can result in greater risk for an AI-

enabled project, so understanding the responsibilities for each of these roles during the requirements process will benefit the project. Users should be consulted throughout requirements gathering to ensure requirements will meet mission needs from a user perspective.

Involving T&E personnel before acquisition can be challenging because funding and adequate resources to enable T&E may not have been identified this early in the SELC. If T&E personnel are not embedded within the program, it is essential for program managers to begin planning for enabling T&E. The earlier the T&E staff are engaged in supporting the AIES, the greater the likelihood that mission objectives will be met.

5.1.2 Prototyping

Creating a prototype (or proof of concept) can validate that AI is needed and is the right solution, while also confirming the proposed requirements. Prototypes can help data scientists determine if sufficient data quantity and quality exist for a viable AI model (see Section 6.1.2.2 for more details on this process). Moreover, prototyping allows for the identification of data needs, data generation and labeling processes, and the responsible parties (e.g., contractors or third parties) for these tasks. Involving data scientists in the prototyping process can provide valuable insights on the potential AI types to consider for a system. It is crucial to evaluate prototypes in a relevant context to yield useful answers. This may necessitate simulating salient aspects of a mission to effectively learn from a prototype. Modeling and simulation may be required to interact with the AI, validate a prototype, and choose the best solutions.

A best practice for implementing AIES is to ensure that the requirements truly necessitate AI functionality. Making the distinction between capabilities that require an AI-enabled solution versus a non-AI solution can be effective in refining requirements. Sometimes, following a simple set of rules in a flowchart can be more effective than implementing a much more complex ML solution. Furthermore, an AIES can potentially evolve after a rules-based system has been implemented initially. A logic-based prototype might meet user needs just as effectively as an AI-enabled solution.

While program staff are typically responsible for ensuring prototyping occurs, T&E personnel should be informed on the results of or should potentially be involved in prototyping exploration. The benefits of prototyping should be evaluated, and results may help with the development of evaluation plans to ensure a rigorous exploration has occurred. Prototyping can be more effective in a virtual/digital representation of the operational environment.

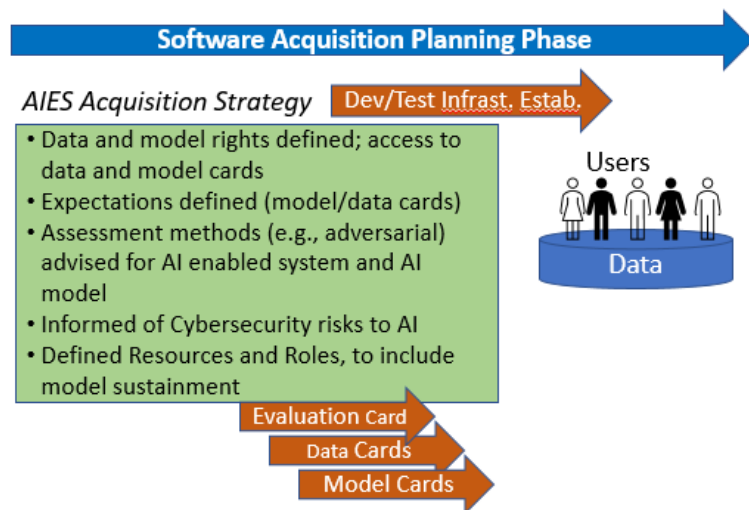
5.1.3 AIES Hardware Infrastructure Requirements

AIES hardware requirements are like those of any other system in that they require system requirements to be clearly provided to define the limits and bounds within which the software needs to operate (e.g., CPU/GPU, memory, storage within a particular Size, Weight, and Power) that can be validated during testing. Infrastructure requirements are necessary for both deployed systems and the model training and data pipelines needed for AIES. Infrastructure for AI typically requires high-performance computing, especially during model development. The large datasets required for an AIES implementation are often cloud-hosted and demand CPUs and GPUs, which are not normally necessary for non-AIES. AI operating in a cloud provides scalability, but in some cases, a cloud solution may not be possible, and then hardware requirements become even more crucial. If specified in the requirements, it will be necessary for

a DoD T&E representative to validate this infrastructure (including its security) as part of the requirements verification and validation.

There is often a difference between the hardware required for training the AIES versus the hardware needed for the inference that the AIES performs in operation. Often, training requires much more processing power than inference—e.g., a model may be able to fit onto a Group 2 or 3 Unmanned Aircraft System to perform object detection or classification inference once it's trained, but it may need cloud resources to train it. Retraining in the lab while the deployed system is in the field is a reason why prototyping can be beneficial.

5.2 Acquisition Strategy for AIES



During the early phases of contracting and proposal development, there are considerations that must be addressed to account for AIES unique performance to enable the most effective test and evaluation to occur.

This section provides suggestions for considerations to include in the acquisition strategy and/or the contracts for AIES. These recommendations may also lead to changes to other related acquisition

products (e.g., Statement of Work, Intellectual Property [IP] Strategies). Key topics covered include:

Data Rights including those related to generated data

Data Quality

Model Rights

Protections for AI/Model Data

Communicating the Relevance of the AI Threat During Contracting

Sufficient Resources to Execute AI Development Program

Request For Proposals

It is recommended DoD conduct algorithm testing and treat the AI model itself as a subcomponent with its own specific T&E requirements. This would then be followed by testing of the overall AIES system. This separation will allow better characterization of the model to include some of its surrounding/enabling subcomponents. Fully characterizing the AI model would then allow the AIES testing to be closer to a non-AIES.

Taking this approach will require language in contracts to have access to test this AI model subcomponent and test infrastructure resources, as well as DoD Subject Matter Expert (SME)/contractor SME to effectively conduct such assessments. As described later in this

document, this approach would ideally be enabled by independent tools/automated testing producing independently verifiable results as the models are trained. It is acknowledged that this approach may not be feasible, but when it can occur it will greatly reduce the complexity of testing an AIES, shortening the time to operationalize.

The AI model will continue to evolve once deployed as it learns from the environment it is deployed within. This non-static nature requires the need to not only track the model's performance but also to have an ecosystem, similar to where the model was originally trained, where the model can be brought back for retraining. Afterward, the improved model would be redeployed. This iterative nature requires access to SMEs and resources that the contracts for AIES need to plan to address.

5.2.1 Data Rights

Understanding the data used to train the model can help inform the known edges of the AIES performance, allowing the performance beyond those ranges to be fully characterized, ideally validating built-in guardrails and AIES' self-reported confidence scores.

DoD contracts have historically been based upon Federal Acquisition Regulation and the DoD specific Defense Federal Regulation Supplement , which generally included strict rights to data, software, and information, giving DoD fuller control of the systems they were procuring. More recently, other contract vehicles have been used, such as Other Transaction Authorities, with the goal that simplified contracts with less onerous terms would attract greater innovation from commercial entities and potentially speed capability delivery. These contracts could be insufficient for AI-enabled capabilities if they do not specify rights to data. Specifically, contracts should seek data to assess the processes used to train and test AI models that enable the AIES. This is especially true for cases where DoD does not intend to provide any or limited amounts of model data and has put the burden of data sourcing, labeling, etc. largely on the contractor providing the AIES. In general, there are three main approaches related to model training data that DoD should determine and consider in contracts:

- Buy “off-the-shelf” pre-trained models

- Provide government-furnished information (GFI) for the contractor to train models

- Pay contractor to gather data

In all three cases, there is a need to plan for handling periodic model retraining post deployment. This may entail creating new data sets and/or collecting data that the model uses and/or produces as it operates (perhaps as enabled by the broader AIES). This retraining will be triggered by model performance changes (e.g., drift) that is detected by the AIES or the user.

Determining “who” and “how” the data will be made available to design and train the AIES needs to be explored and specified in the contract for DoD systems. For example, data may be available from internal resources, external publicly available resources, and/or other external licensed sources. If proceeding to purchase licensed data, besides the capital cost of acquiring initial datasets, any continuation of data consumption/collection required post the production deployment to keep the model relevant and monitor model drifting will increase labor costs. Increased labor cost is related to required support for data labeling, data splitting, and/or generating synthetic data, which is usually underestimated. In all cases, the DoD should be clear about what the responsibilities are and the short/long-term implications. Costs to protect the data,

understanding data provenance, and supply chains (e.g., attack vectors on public or purchased data sources) should be accounted for to prevent under-planning and to avoid spillage or even data poisoning. These plans should also ensure data sufficiency and that the data is sufficiently parsed to be used for the discrete phases the model will go through (training, and various phases of test and integration).

From a contracting perspective it is necessary to ensure the government owns and/or has rights to all the data and data transformations that occur to enable it to be used for training the AI model. While these rights may lead to increased contract cost, they can lead to longer-term savings for reproducibility and algorithm adaptation. This DoD ownership will ensure the highest level of control over the model development and rights to the trained model for subsequent maintenance and/or reuse in other applications. Any decision to take this approach should be specified in the procurement/contracts process.

Often DoD chooses to place this burden on the contractor, especially when data and SMEs to manage/label the data are in short supply. Another means to mitigate this lack of skill may be to contract a specialist for the data discretely, separate from the AIES's development. There are Business Process Agreements enabled through the Chief Digital and Artificial Intelligence Office (CDAO) that are specifically intended for this purpose, to enable DoD organizations to have ease of access to data enabling teams. In all cases the government will need rights to the data if they plan to use it for any model retraining.

Formats of living data are an important part of the negotiated delivery of the data. Ensuring there are effective data pipelines and proper deployment/accessibility via technologies such as containerization should be specified to ensure effective and continued use.

5.2.2 Data Quality

Data quality for the data used to train the AI model drives the success of the AIES. Data quality encompasses many different aspects of the data utilized in ML projects, including accessibility/availability, accuracy/correctness, completeness, consistency, credibility, relevance, timeliness, and more. It is important to define data quality in the contract if the DoD does not have direct control over the data quality and is dependent upon this being ensured by a vendor.

It has been an industry best practice to adopt Data Cards and Model Cards documenting the contents of a database or specifications of trained ML models. This method enables a clear overview of primary information about the dataset and allows development teams to understand the purpose of the dataset and further access the quality of data, including the following basic items:

- Data origin, period of data collection, intended dataset usage, and relevant timeframe for the dataset utilization if any
- File source, size, licenses, and data fields
- Storage location and frequency of updates
- Assumptions and limitations of the dataset
- Data sanitization method for the use of Personally Identifiable Information/Personal Health Information (PII/PHI)

- Potential privacy and ethical risks, especially related to the PII/PHI data
- If the data is labeled, the methodology for labeling the data and labeling validation, if applicable

Data Cards are further described in Section 6.1.1.3 Data Cards should be an expected deliverable specified in the contract ensuring that they are produced and delivered by the organization(s) that owns the responsibility to produce the data and ensure its quality.

The source of the data is an important consideration for ensuring the contract allows for the resources that may be needed to validate data quality. Data that is derived from operational environment should be validated to ensure it truly represents the operational considerations on employments. Data that is generated synthetically also needs this operational validation. Due to the nature of synthetic data, resources may need to perform this operational validation several times as the synthetic data is adjusted to optimize its operational relevance. In all cases the vendor/government needs to maintain the Data Card to ensure it is updated as the data changes through the process to ensure T&E can track what is mapped to the AIES.

5.2.3 Model Rights

The DoD should approach AIES with agreements to share the nature of their technology, so that the system is not fully black boxed to certain government representatives. This should include an understanding of the underlying model architecture, algorithms, and/or weights that are enabling the system so that effective T&E can be properly designed, and associated risks and vulnerabilities understood.

An AIES that the government pays to be trained and enabled has a high value and could potentially be resold to another U.S. government agency or to another commercial or foreign organization. The AIES, the trained model(s) within the system, the training process, or the training data may be of high enough value or security concern to warrant contracts to prevent reuse beyond the stipulated contract. These stipulations will be useful for T&E organizations (especially red teams) who need to determine the level of risk for the fielded systems.

5.2.4 Protections for AI Model Training Data

Early access to relevant and operational data is critical to the successful development, testing, and adoption of ML systems. In preparation of data to support AIES development, it is essential to have data usage agreements, memorandums of understanding/agreement, or other data access controls in place to allow proper access to the datasets from various sources and roles.

Additionally, periods of access and/or allocations to all designated teams including development, validation, and testing, and even sharing with potential external partners may be required.

The following are some core elements required to be addressed in any form of data ownership agreement/memorandum :

- Data Management
- Data Location
- Data Access
- Data Security

- Data Rights and Retention
- Data Provenance

An ML program/project should also consider data encryption, privacy, a necessary data destruction process, and also refer to specific regulations and laws (e.g., the General Data Protection Regulation and the California Consumer Privacy Act of 2018, especially regarding personal data such as PHI/PII), if applicable.

Data that is Controlled Unclassified Information or classified should be stipulated up front to ensure that the appropriately cleared personnel and systems are in place to support the program. Additionally, classification concerns with data aggregation should be considered.

5.2.5 Communicating the Relevance of the AI Threat During Contracting

When the DoD is preparing to engage in an agreement to procure an AIES they should establish up front the assumptions on the relevant threats that the AIES is expected to encounter, continue to operate with, and/or react to in a specified way. There are resources available to discuss this emerging space in terms of ontologies and possible threats (see Appendix I. Taxonomy for AI Cybersecurity and Common Threats to ML). These should be considered and used to produce more specific guidance that is relevant to the specific AIES.

Program Management Offices (PMOs) and DoD test agencies use the Validated Online Lifecycle Threat (VOLT) and similar threat documentation in order to document the known threats the AIES will have to withstand to successfully operate in its planned environment. It is not specifically required, but the VOLT should be used as a common vehicle to capture specific AI relevant threats to the system that the organization plans to employ. As defined in the AIES contract, the VOLT can be leveraged by:

- DoD PMOs in order to ensure the system's intended use is relevant and risk to contract for a planned system/upgrade is manageable.
- Commercial developers employ methods to mitigate.
- Testers help design, plan, and build effective testing strategies and resources (data, SME, tools). This information would also be used to inform the conduct of Cyber Table Top exercises and Mission Based Cyber Risk Assessments.
- Sustainment organizations to be fully informed and ready to handle the expected threats (training, mitigation, and handling instructions).
- Authorizing Official regarding the risk to the system's security accreditation as informed by the Risk Management Framework.
- Teams developing or managing separate systems that interface with the AIES.

5.2.6 Sufficient Resources to Execute AI Development Program

Contracts need to account for and explicitly state expectations for the resources that will need to be provided for AIES. This will ensure the funding aligns to expected project execution, averting the risk of cost overruns and delays from unplanned and unexpected events. The T&E of AIES will require more resources than for a non-AI enabled capability and proper planning is needed

to ensure the program plans for these increased costs and takes the steps to ensure necessary activities occur.

- **Data:** The critical resource to help ensure that all data considerations are addressed is one of the most critical resources for AIES. This is influenced by who will be responsible for generating, collecting, and curating the data. It encompasses the resources necessary for data storage, governance, and maintenance. Tools that can provide the independent validation of these activities will reduce the costs of later complex T&E. Additional resources will be required for adversarial AI testing when data and/or models are externally acquired/gathered for curating mission relevant adversarial conditions the AIES must perform under. For all AI, there should be sufficient data generated so that training and test data are separated. If the vendor is providing their own training data, the DoD will need to have a contract with a different vendor to provide independent test data set (not a subset of the training data the vendor already had).
- **Human Machine Teaming:** AI involves enabling human actions and humans need to be part of the early processes to ensure this human machine bond is well aligned. Developers will need access to users. This can be a challenge for capabilities with users already in short supply. If users cannot be made available, the risk of successful user acceptance in later T&E greatly decreases, leading to suitability failures due to lack of trust in the AIES. Agreements on the level of user support and any costs to do so should be specified.
- **Model Training and Sustainment:** The infrastructure to enable the AIES through the full model training and integration as well as the tools and infrastructure for data/data pipeline need to be available through an ecosystem available to the AIES development team. This needs to be defined in the contract to ensure it is available at the time it is needed to address the model retraining or allowances for redeployment in different operational conditions (requires retraining). This environment needs to accommodate tools that provide insights on model performance and automated outputs that can produce independent evidence on the model training.
- **More resources are needed to execute AI's full lifecycle** than may be expected, as it involves a longer tail and higher resources than traditional software and has a higher dependency on user engagement. The AIES will need to be monitored and pulled back for adjustments if any hyper parameters are set to continue to self-train and evolve, or data drift occurs causing the AIES to operate beyond where it was originally trained to perform. As this is almost certain to happen, there should be considerations built into the contract or contract clauses that clarify how these will be handled. This may include stipulation on availability of vendor SMEs, data, development pipelines, and the time, schedule, and relevant testing considerations. To facilitate these activities, the AIES will need to have built in feedback loops, data collection, and logging necessary to best handle these scenarios and prevent long drawn-out contractual issues that may lead to the AIES's early retirement.
- **Changes to Intended Use:** Contracts may include clauses to address additional or added scenarios where the AIES is expected to be deployed in conditions beyond its originally intended use. Some types of AI (e.g., target recognition) can have many

employments (drones or other platforms), leading to more application of an AIES, perhaps beyond original design. Employment changes may involve ensuring the AIES can be hardened to withstand emergent threats. When this new employment changes the original use case(s), the AIES will require new data and rigorous T&E. These contract clauses should include these necessary resources.

5.2.7 Requests for Proposals (RFPs)

DoD will typically develop RFPs to describe a capability that DoD plans to acquire/developed. Commercial vendors will respond to the RFP, describing details of their solution, enticing DoD to choose the solution(s) to fully contract with. In many cases, DoD's RFPs may not specify the need for vendors to provide AIES however, it is highly likely that vendors will provide AI-enabled capabilities to attempt to provide the most competitive RFP response.

Due to this high likelihood of AI's presence in future commercial solutions, DOD should ask the contractor if AI is present in their system. If due to IP reasons, this is not answered clearly, then the **RFPs should assume the system *will* be AI-enabled** in some capacity. DoD should require that respondents advise of the presence of AI, the type of learning, and the nature of its implementation (model type and seek enough information about the AI to enable the RFP evaluations with a Model Card). The presence of AI does introduce complications to T&E and so it is crucial that RFPs seek for vendors to identify its presence.

If the RFP evaluation includes a demonstration, it would potentially require DoD to provide data to enable the vendor to train the AIES. This support could include DoD providing a characterized training dataset (see 6.1.1.3 on Data Cards). This process may benefit from engaging a T&E SME to advise and support the evaluation to help them make the most informed decision to include the DoD's ability to fully characterize and evaluate the proposed solution for rapid delivery.

5.3 Test Strategy

This section discusses the T&E preparations of the AIES focusing on essential test planning, considerations, and required artifacts that should occur. The following topics are covered:

- Test & Evaluation Strategies and Test & Evaluation Master Plans (TEMPs)
- Test Planning with T&E Workshop
 - T&E Roadmap and Checklists (details in 7.4Appendix J)

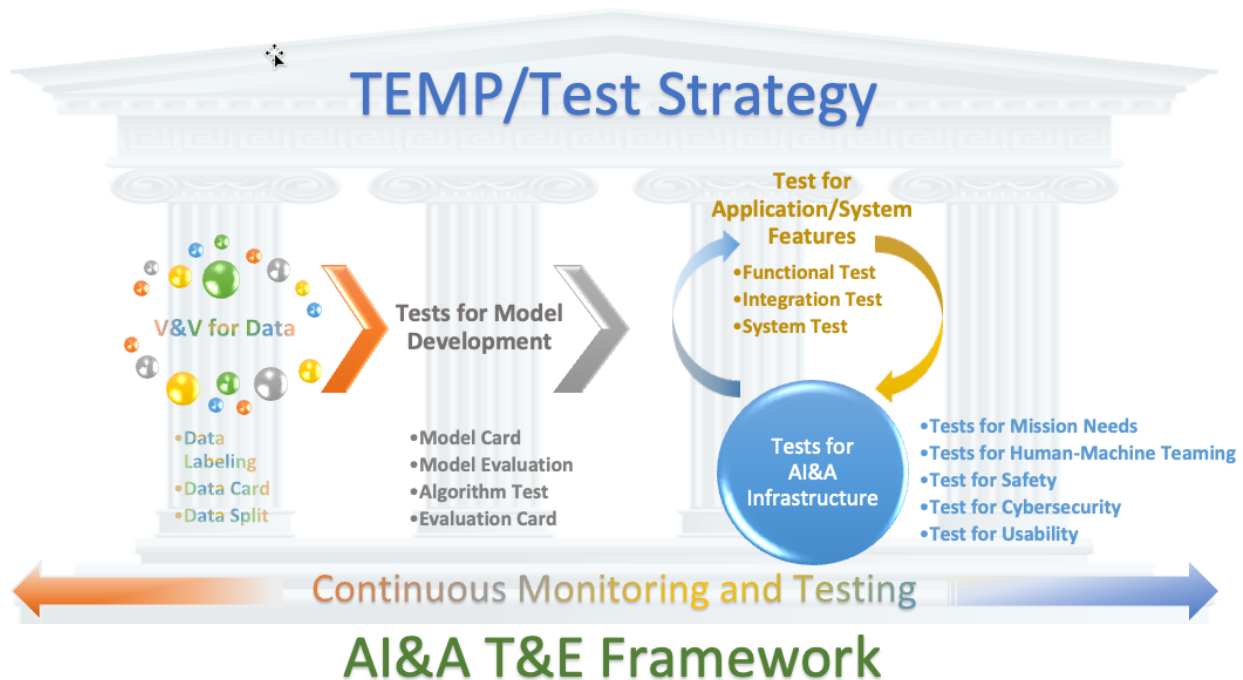


Figure 5-3. AI/ML T&E Framework

Since ML systems differ from traditional SW systems, they require changes to the T&E process that should be captured in the relevant documentation artifacts. For example, in addition to tests that look at specific software applications/system features and data, AIES will require T&E to evaluate the model, the process to train/training data, sufficiency of the infrastructure (e.g., GPUs) to enable it, and to store any logs. Successful ML systems should take into consideration the scope and goals of the operational mission that the system intends to meet and human-machine teaming techniques that can help to ensure adoption as well as safe and effective use of the planned system. These topics are illustrated in Figure 5-3.

CI/CD require, by their very process, the iterative testing of systems throughout their lifecycles. After deployment, continuous AI performance monitoring into production should also be planned as applicable.

If agile software development methodology is employed, T&E team should ensure the above testing and reporting is properly built into the methodology. The software development and testing and associated reporting and correction should also be tailored to adopt industry standards for agile development.

To prepare to sufficiently test ML enabled system, the T&E team needs to tackle the following challenges:

- Determine testing requirements when state space size makes testing all cases infeasible or the open world problem makes enumerating all cases impossible.
- Address the potentially invalid assumption that these emergent systems can be decomposed.
- Deal with dynamically varying systems that are potentially never in a “final” state during deployment.

- According to Dr. Freeman’s Best Practices for Addressing New Challenges in Testing and Evaluating AIES, (Refer to 7.4Appendix C for details.) These themes are key considerations for the T&E team to address when building the T&E strategy document.

5.3.1 Test & Evaluation Strategy/TEMP

DoD T&E strategy (and equivalent documents like the Test & Evaluation Master Plan) outlines the test strategy for the overall program, and it should adequately describe the overall test strategy objectives, scope, rationale, schedule, and resources. The T&E strategy is updated to support acquisition milestones and significant program changes and captures the current test and evaluation strategy, which can change as additional information is discovered.

The purpose of the T&E strategy is to credibly assess the technical requirements and operational performance of the program and serves as the overarching document to guide the ML T&E process. For AIES, the T&E strategy should:

- Define the resources and cost to establish the T&E Framework supporting AIES into the program’s overarching test strategy.
- Outline AI-specific T&E products required to execute the T&E strategy (e.g., T&E strategy development, use cases, safety certification plans, Data Cards, Model Cards, Evaluation Cards).
- Specify Verification and Validation (V&V) requirements and test data needs.

A T&E strategy that includes an AIES should consist of three major sections: (1) Introduction, (2) T&E Framework, and (3) T&E Resources. This basic organization ensures comprehensive coverage of key issues assuring effective AI T&E in a logical order and sequence. Refer to the [Test and Evaluation Enterprise Guidebook, August 2022](#), the [T&E Strategy DoD Manual \(FY23\)](#), and [T&E Strategy Companion Guide \(FY24\)](#) and [Artificial Intelligence and Autonomy \(AI&A\) T&E Focus Area Chapter \(FY23\) under DOT&E/DTE&A joint efforts](#).

The T&E strategy should capture sufficient knowledge across the stakeholders of AI's unique challenges and requirements for the system and will set common expectations of how to plan, execute T&E activities, and collect essential evidence to assess the AIES for specific projects. Refer to the [JAIC \(transitioned to CDAO\) TEMP Template for a Program with AI_20220114](#), which provides a framework and guidance to assist project managers, system developers, testers, and operators for test and evaluation activities required to establish and engender trust in the ML fielded system.

T&E strategy describes the T&E team’s responsibilities of overseeing all tests and compiling test results. Testing for the product shall be done at all stages of development by model vendor, system integration contractor, or government personnel. Performance of AIES or applications shall be evaluated at a granular level to carefully look at key stages of the process to evaluate the AI-enabled product incrementally to draw fuller conclusions on its performance. This should include assessing tool/algorithm, system integration capabilities, human-system integration, and operational performance. Government teams should ensure that contract and T&E strategy has specified roles and responsibilities to deliver required T&E, test artifacts/results, analysis and evaluations, and supporting test data for DoD T&E authority review, approval, and use.

Test Strategy should describe a top-level framework and process flow of T&E activities planned for different development stages. Typically, it should include testing and/or V&V of the following artifacts throughout the SELC:

- Data Labeling and Data Card
- Model Test and Evaluation
 - Model Card
 - Algorithm Test and Metrics
 - Functional, Integration and System Test and Metrics
 - Evaluation Card
- Operation Test (OT) and Metrics (to include model performance assessment on intended hardware/operational environment)
- Human Machine Test and Metrics
- Cyber Security Test and Metrics
- Usability Test and Metrics

Model development methodology and data used to train models must be tested and validated/evaluated through Data/Model Card submission, algorithm test, etc., before system integration. Testing, tuning, and model evaluation is required. If applicable, test strategy should specify if the ML-enabled system will undergo continuous development and evaluation for improved model accuracy using outcomes from routine data pulls.

More details can be referenced in [DoD Instruction 5000-87](#), *Section 3.2 f: Test Strategy*.

5.3.2 Test Planning with T&E Workshop

Test planning is the most critical element of T&E to ensure any successful test, evaluation, and/or assessment activities required for AI-enabled product and/or systems. This planning would begin as early as the Requirements during prototyping and build into the Acquisition Strategy where the initial draft Evaluation Plan is strategized and developed.

As recommended by the CDAO T&E framework, it is an effective practice to formulate a T&E Workshop to initiate the planning with all involved teams, especially the mission user groups. The goals of the T&E workshop are to achieve the following through a set of group exercises focusing on each of the four primary areas including AI Tool/Algorithm Testing, Human-Machine Testing, System Integration Testing, Security Testing, and Operational Testing:

1. Identify AIES requirements and better understand operational needs.
2. Develop T&E strategy and document high level planning information in T&E strategy.
3. Update T&E strategy as mission requirements and operational environment evolve.
4. Start coordinating test planning and supporting details (see 7.4 Appendix J) with involved partners and stakeholders.

5. Work with model development teams of whole systems to disclose any components reliant on developed system, which can potentially cause unplanned or unexpected AI behaviors in the AIES.

With the four primary test areas, it is critical to understand operational needs and map them into testable requirements throughout the T&E Workshop/working sessions and capture them into the T&E Strategy and developing corresponding Evaluation Cards to support downstream test planning. During the test planning stage, T&E workshop should also define required performance metrics and establish processes (e.g., Evaluation Card in 6.1.1.1) to provide consistent reporting of test results. This will ensure a standard approach to produce comparable T&E results among different model vendors, and further leverage in-depth analyses of common/custom metrics.

Refer to workshop details in the JAIC (transitioned to CDAO) T&E Workshop . Group exercise examples recommended for the T&E Workshop are included in 7.4Appendix L.

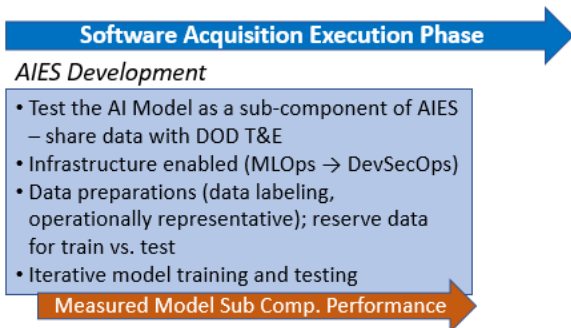
6 Software Execution Phase

The following sections discuss the execution phase and the activities as well as the resource/time considerations to enable the test and evaluation of this system by Development Test & Evaluation (DT&E) and Operational Test & Evaluation (OT&E) organizations.

This section covers the three phases of the acquisition lifecycle, which is consistent with the USAF's Artificial Intelligence Acquisition Guidebook:

- AIES Development
- Testing of the AIES
- Deployment/Sustainment of the AIES

6.1 Development and Model Testing



The Development Phase includes the development, testing, and training of the AI Model as well as the development of the AIES. During this phase, the team will leverage and update the Evaluation Card and the Model and Data Cards. These resources will greatly enable the T&E community to understand the AIES and use this toward the total body of test evidence. The teams must take care to treat the data with proper protections including reserving key data that is used for model testing for record.

6.1.1 AIES “Cards”

There are three cards that are used to capture key information about the AI model, its training, and the model(s)' evaluation plan/results that will be useful for the T&E of the AIES. These templates enable the DoD programs, development teams, and test and evaluation teams to document and organize all aspects of the model: from training data, metrics, and expected model behaviors to results.

Model Cards describe aspects of the model, its purpose, and its development. They will be generated and updated regularly during development cycles, possibly through automated tools. Data and Model Cards must be reviewed and verified by the T&E team throughout the SDLC process and any associated Evaluation Cards must be updated to reflect any changes made to the data and Model Cards over time.

These cards should begin during the earlier planning phases (requirements, acquisition strategy) and be updated frequently throughout the SDLC.

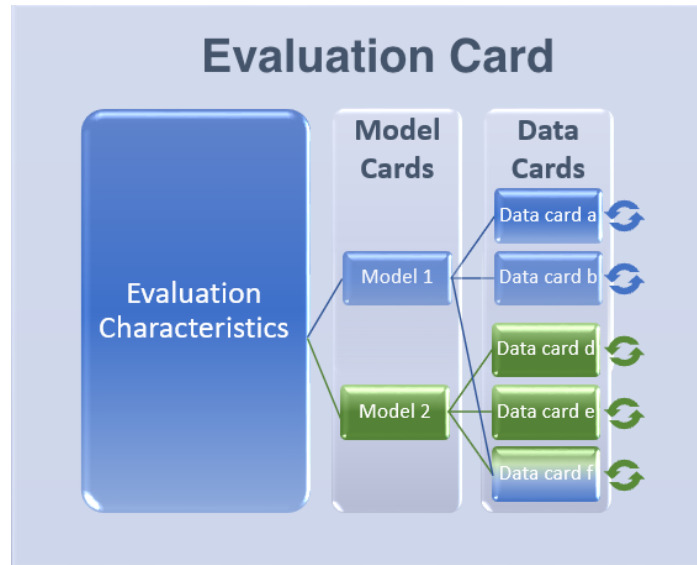


Figure 6-1. Notional Card Linkages

The acquisition strategy/contract for the AIES should discuss (1) the need to produce each of these cards; (2) who should create them, and when these artifacts are provided; (3) under what circumstances they are updated, and (4) the limits of any information sharing and management of this data.

6.1.1.1 Evaluation Card

An AI model's Evaluation Card documents the methods and conditions under which the AI models will be evaluated and tracks results of these AI-subcomponent level assessments. The Evaluation Card approach ensures a consistent and well understood T&E methodology is used . The Evaluation Card would be developed initially during the Acquisition Strategy Phase, leveraging information resulting from the prototyping (5.1.2), data exploration (6.1.2.2) and any test planning workshops (5.3.2).

Test plans are a more traditional test artifact, but AI model testing is more complex and warrants this additional specific documentation found in Evaluation Cards. There may be a need for multiple Evaluation Cards throughout the engineering lifecycle. For instance, an overall system evaluation as well as component level evaluations performed during development and integration. These cards could be included or referred to by the AIES test plan. Additional planning and characterization need to occur to facilitate an independent evaluation and this plan needs to be established and agreed to by all parties. The Evaluation Card provides an organized checklist that specifies this plan and guides the T&E process, informing on the scope, technical rigor, and serving as a roadmap to guide the T&E community. Both data and Model Cards will be linked to the Evaluation Card providing traceability of the process that built the AI model that enables the AIES. Refer to details in Appendix D. Evaluation Card Template.

Additionally, Evaluation Cards can be used to report on the results when monitoring a system in deployment and track important behavior such as data drift. The cards will provide teams with the information needed to retrain the model to sustain performance. It can also be used to help meet emerging needs or to tune performance expectations to reflect new requirements.

T&E organizations should maintain a database of Evaluation Cards for T&E professionals to use. It can serve as a library of previous AI model evaluations to inform subsequent efforts.

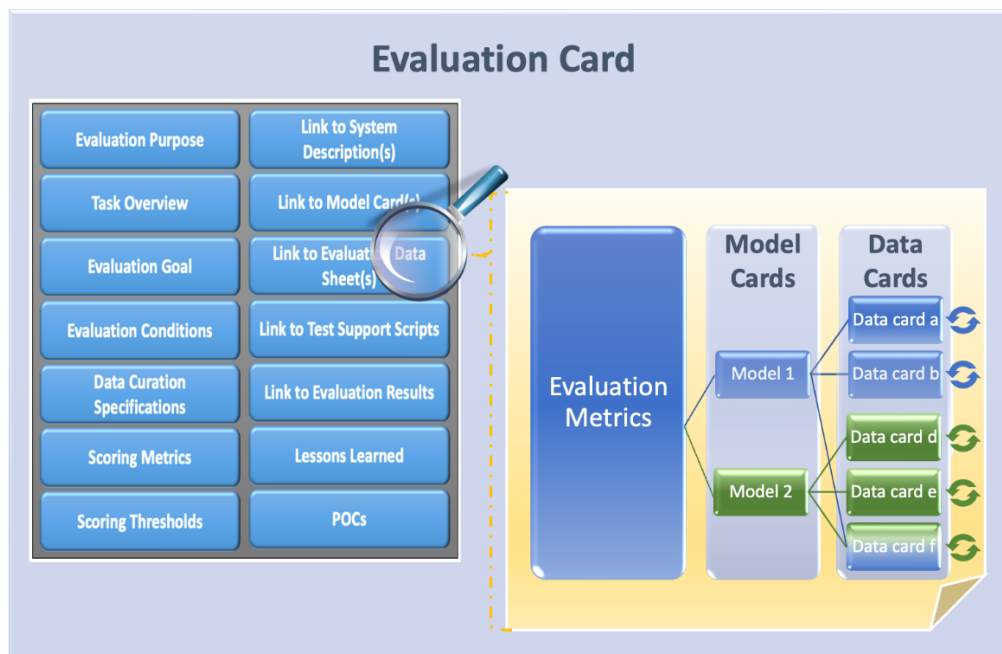


Figure 6-2. Primary Elements for the AI T&E Evaluation Card

Refer to roles and responsibilities mapping for Evaluation Cards task in Section 5.

6.1.1.2 Model Card Development

ML model development is a training/test process that is inherently iterative. Model development starts with data and data characterization, feature selection, feature weight selection, and algorithm selection. A good model card is easily consumable by system developers, data scientists, and T&E staff alike. Model Cards should be semi-automatically generated from the training process. Lastly, Model Cards should capture the AI model's operational requirements, especially as they change over the life the AI model.

Any AI/ML libraries exist with models that have been trained with baseline level performance. These models can be modified to meet domain needs by modifying the preprocessing of data to map to the feature space, feature engineering, or data manipulation due to computational load balancing and the (hyper)parameters used by the ML algorithm. After this model training, it is tested to determine if it is close to the required level of performance. It is likely this will take more than one iteration and the model will go back for more training to improve/adjust the feature weights, feature selection, or even algorithm choices are adjusted, during model retraining. The time to train a system to meet the required accuracy levels cannot be accurately predicted in all cases, particularly for more sophisticated ML models, such as deep learning.

Because of the iterative nature of AIES development, it is essential that each training run is adequately logged and documented so that the development and T&E teams can track changes in performance as a result of variations in preprocessing, feature engineering, weighting, and parameter adjustments; Model Cards specify details of the training process.

Model Cards can provide diagnostic information for future potential issues. For example, if a model were trained/tested on a single or narrow dataset it can result in overfitting, which is when the model trained reflects the training data to a level that the model can fail on new data that differs from the training/test set. Refer to details in 7.4Appendix E.

Under an automated MLOps framework, a model card could be generated semi-automatically from training logs for each training iteration. For instance, open-source libraries facilitate the generation of training logs which can be parsed to help build Model Cards, semi-automatically, reducing the documentation burden on the developer and creating a more independent set of model training as test evidence. Model Cards are associated with the appropriate Evaluation Cards to enable overall T&E practices.

6.1.1.3 Data Card Development

Data Cards document the specifications of the datasets that are potentially used to train an AI model and are key to understanding the scope, quality, and pedigree of these dataset(s). Data Cards are specifically designed to capture composition, collection process, recommended uses, collector's motivation, etc., for acquired/collected dataset. Data Cards have the potential to increase transparency and accountability within the ML community, mitigate unwanted biases in ML development and systems, facilitate greater reproducibility of results, and help researchers and practitioners select more appropriate datasets for their chosen tasks .

In 2019, Google, Microsoft, and members of academia released two papers proposing Data Card format and structure to document the assumptions and limitations built into datasets and models. Google adopted this format and is implementing it in their cloud AI department. Google also released the Model Card Toolkit (MCT) built in the JavaScript Object Notation (JSON) format to promote Model Card use across the industry . The MCT served as the foundational framework leveraged by CDAO when it established templates and started to champion for the adoption of the Data Cards and Model Cards in 2020. (Reference these templates in 7.4Appendix F and 7.4Appendix E.) DoD/CDAO are continuing to develop pertinent standards and guidance for Data Cards and Model Cards.

Data Cards should be developed during the data collection phase and updated regularly throughout the development cycles as new datasets are acquired or existing datasets are updated. Additionally, listed data source, datasets, and linked Data Cards should also be updated accordingly in the project managed Data Manifest. Implementing a management process for both Data Cards and Model Cards will enable the mechanisms to achieve data traceability and explainable model development. Refer to roles and responsibilities mapping for data and Model Cards tasks in Section 5. Refer to details in Appendix F. Data Card Template.

It is highly recommended that all ML developed products have associated Data and Model Card(s) to assist the T&E process. Understanding the operational context that is reflected in the training data and the trained model informs on the edges of performance that may need to be explored in model testing and ultimately scope the later AIES testing.

6.1.2 AI Model testing during the Development Phase

During the Development phase, the AI model will be matured, tested, and retrained as a subcomponent of the overall system. When testing organizations can review Model and Data

Cards and the evidence they provide, testers will be better able to assess the rigor of this process and the quality of the data that was used to build the AI model. This will yield results that are useful test evidence into the overall maturity of the AIES.

One can define the appropriate metrics used to assess the AIES's model by starting with industry standard metrics approaches (e.g., confusion matrix), but consideration must also be given to the nature of the users preferences of the system to show which metrics are truly aligned to users' needs (ideally as stated in use cases).

Understanding the robustness of the use cases for the AIES can also start to inform on the ethics of the AIES by helping us to understand the limits of performance that should be validated during T&E. Understanding the rigor of the protections (especially those ideally stipulated in contracts) which are enabled through security policy and or infrastructure information, is crucial for assessing AIES cybersecurity risks.

This open approach of evaluating the AI subcomponent may not be feasible for all AIES and AI models. Intellectual property may cause the AI development processes to be black boxed. This type of system will increase the needs to further explore the system during the Operational Testing Phase (6.2.3).

6.1.2.1 General Paradigms within ML Model Development

There are four primary technical approaches commonly employed during ML/AIES development:

Supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. Supervised learning techniques are conventionally applied in use cases for which all reference targets in the training data are already known and validated before model training is performed. Supervised learning therefore typically **requires manual annotation of curated data to align inputs with their expected outputs**.

Conversely, unsupervised learning methods are suitable for exploratory tasks in which targets are unknown prior to model training. Consequently, unsupervised learning **requires additional expert review** to validate the model's target predictions during training, which is repeated over many iterations as these models are re-trained with additional feedback from domain experts and members of the development team.

Semi-supervised learning combines supervised and unsupervised learning methods by including **some degree of human intervention** and review during the development process for the subset of training instances which lack pre-validated targets.

Reinforcement learning adopts an agent-based approach for optimizing policies given a specific goal; these policies are informed by automatically performing various sets of actions over multiple steps and evaluating their relative reward—that is, the effect which these actions produce towards achieving the desired goal.

Mission-oriented systems traditionally employ either supervised learning or reinforcement learning models except in use cases where data exploration is an inherent process imposed by conditions of a particular task.

The T&E of each of these models will require test data that was separated out from the model training data. Some will require expert review, which can be extensive if testing drives retraining

and additional testing. Testing needs to plan to include the appropriate SMEs to validate system performance, especially in cases where no ground truth is well defined and curated predictions on outcomes is available to leverage.

6.1.2.2 Exploratory Data Analysis and Data Profiling

The development of every AIES must be informed first and foremost by the specifications and characteristics of the data it is designed to serve. While data scientists can leverage various automated and semi-supervised techniques to lead the data profiling effort, a comprehensive characterization of the data must ideally be performed and validated with guidance and inputs from domain experts in a collaborative process. This exploratory data analysis (EDA) phase enables the AI development team to identify key technical requirements of the system. For example, a multi-label prediction task (in which individual instances can belong to more than one class) or a multi-modal dataset (in which features are comprised of assorted data types such as text, video, images, and audio) may require an ensemble-based architecture to integrate separate specialized models trained on distinct subsets of classes and features Table 6-1 below provides some important dataset characteristics and conditions to consider during the EDA stage.

Table 6-1. Common Dataset Properties and Data Profiling Questions

Support Role	Dataset Properties	Questions
Data Scientist	Diversity	How many unique values exist for each field/feature set?
	Dimensionality	What is the maximum number of known features specified in the dataset?
	Feature Independence	Based on conventional statistical tests, how strongly do different feature (sub)sets correlate with one another?
	Sparsity	What proportion of features have missing/null values? Does the data contain any imputed values from previous preprocessing steps?
	Regularization	Do feature sets require some form of regularization (e.g., case conversion, sampling normalization, etc.)?
	Data Types	What data types are represented among the fields/feature set (e.g., categorical, numeric, string)?
	Multiple Labels	What features (if any) contain multiple categorical labels? What proportion of instances contain multiple labels?
	Modality	Does the data contain a combination of media types (e.g. text, video, images, audio, etc.)?
	Structure	How is the data structured (CSV, SQL table, JSON, etc.)? Is the data party unstructured/semi-structured? Is the data structured in different formats (i.e. data lakes)?
	Readiness	What are the preprocessing/transformation requirements that must be implemented prior to model ingestion?

Support Role	Dataset Properties	Questions
	Storage	Is the data divided between different storage systems (i.e., data silos)?
Domain Expert	Reliability	Based on domain knowledge and experience, what is the known or estimated error rate?
	Domain Coherence	Does the data span distinct (sub)domains? What is the estimated proportion of out-of-domain instances which may require filtering?
	Specificity	Do feature values accurately reflect their targets? Are certain features ambiguous or underspecified?
	Sample Variance	How well does the data reflect the known variance/scope of the domain?
	Provenance	Can elements of the data be associated with their original source/collection method?

Virtually all AI systems involve some level of data preprocessing to prepare raw data for model training and subsequent predictions by trained models. Although the EDA stage addresses some preliminary preprocessing considerations (e.g., regularization, structure, readiness), some additional extraction, transformation, and loading (ETL) process may be required prior to ingestion. These preprocessing subtasks may either be performed in a separate component of the pipeline or handled natively by some AIES, but in either case, the evaluation and integration team should identify and document the data preparation methods that are best suited for their particular use case and ensure these methods are available for operational use. Ideally in all cases, the appropriate personnel (developers and testers) will be able to review the pre-processed data to understand its impact to the model's performance.

The EDA and data preparation stages of the MLOps cycle should be repeated to prevent common problems that may emerge after AI deployment. One such risk is domain drift where the data inputs can change in ways that significantly deviate from the trained model whether through input source changes (e.g., new cameras in computer vision) or natural evolution of the data (e.g., natural language change). Another risk is through the increasing use of generative AI methods which rely on continuous update or self-learning. Domain drift when used in categorization systems is easier, yet not trivial, to detect post-deployment. Generative systems may adapt in a way such that the output does not reflect the goals of the system. This is harder to detect and introduces complexities on sustainment T&E. No matter the source of risk, additional tracking and testing will need to include tracking inputs, results, and system behaviors in a way that allows T&E and analysis post-deployment.

Data profiling should be a top priority when new data sources are introduced into the AIES and when analysts and other users become aware of changes in the problem space, mission, or the reliability of model outputs. Post-deployment reiterations of data profiling tasks should be routinely automated by utilities of the continuous monitoring component within the AIES environment; specifically, OT staff should leverage these utilities to compute metrics like variance, skewness, mutual information, and entropy (among others) to detect anomalies and

distributional shifts in near real-time as operational data in upstream ETL dependencies before ingestion by the AI model . Such automated routines eliminate the need for manual or scheduled data quality and data characterization tests and help largely reduce the risk of unchecked model performance degradation over the course of the AIES lifecycle .

When building infrastructure and architecture to support data science for AI-enabled development, data exploration and visualization is another essential capability to implement as part of the data preparation/analytics platform.

Gartner listed data exploration and visualization as one of the 15 critical capabilities for data science and machine learning platforms, and defines it as follows:

“The ability to visually manipulate, interact with, and explore the data, and perform basic descriptive statistics and pattern detection via a wide variety of exploratory steps and visualization options.

The descriptive and diagnostic analyses included here are the foundation of data science, enabling the user to diagnose the data, explore correlations and start to form hypotheses.”

For consideration of implementing the Data Analytics Dashboard, Gartner also evaluated 16 data science and ML/AL platform vendors providing tools offering 15 defined capabilities, including the Data Exploration and Visualization within the research paper referenced above.

6.1.2.3 Data Preparation and Infrastructure

The development and integration of any AIES must also support data preparation throughout the AI development lifecycle. Data engineers should establish a well-documented data processing framework to account for the following:

Data Governance: The process of managing data performance and compliance to guard against data bias, mitigate bias where possible, and ensure availability, usability, integrity, and security of data in AIES.

Data Sharing Framework: A systematic approach and structure to address the challenges, legal agreements, and requirements needed for managing trusted, secure data sharing.

Data Architecture: The process of ingesting, storing, organizing, and maintaining the created and collected data by an organization, including the lexicon, ontology, security, and privacy standards.

Data Security: The process of ensuring that data is protected and secured to reduce ML risks such as adversarial attacks. This includes safeguards and detection for data poisoning, data configuration management, data access limitations and logging.

Refer to MITRE’s AI Maturity Model and Assessment Tool Guide for more information.

One of MLOps’ best practices is to provide end-to-end visibility for data extraction, model creation, deployment, and monitoring for faster processing. To achieve this, AIES projects should establish and invest in the necessary infrastructure to support data preparation and data sharing processes. Additional customization of data pipelines enables data transformation prior to model training. If applicable, programs should adopt data fabrics within their ecosystems to facilitate end-to-end integration and provide seamless access across multi clouds, data centers, and edge systems. Gartner defines data fabric as a design concept that serves as an integrated

layer (fabric) of data and connecting processes. Data fabric uses continuous analytics over existing, discoverable, and inferred metadata assets to support the design, deployment, and utilization of integrated and reusable data across all environments, including hybrid and multi-cloud platforms. Gartner also lists Data Fabric as a “Top strategic technology trend for 2022” and predicts that by 2024, 25 percent of data management vendors will provide a complete framework for data fabric—up from five percent today.

6.1.2.4 Training and Re-Training

The MLOps cycle emphasizes the importance of continuously training, retraining, and tuning operational AIES to maintain model optimization. ML model training and evaluation cycles are commonly comprised of a series of interdependent subprocesses, including data splits, batch training, and subsequent model updates either by conventional fine-tuning or ensemble methods. The specific configuration of each of subprocess should be determined on a case-by-case basis according to various constraints, including dataset volume, class distribution, dimensionality, hardware support, and domain drift. When training and evaluating different types of models in distinct architectures, it is critical that testers select a common baseline set of training configurations to maintain stable test conditions and ensure a valid comparison between models.

First, all ML model training scenarios require that the data be divided into smaller samples known as data splits for learning and testing. Most training scenarios employ an additional validation sample to further tune the hyperparameters set during the training phase before testing occurs. Figure 6-3 describes the most common data split strategy to assign 70 percent of data instances to the training set, 10 percent to the validation set, and 20 percent to the test set. Generally, testers should perform random sampling without replacement to ensure that there is no overlap in instances between splits to avoid bias. Testers should also adopt a stratified sampling strategy to maximize the likelihood that each data split represents the same class distribution of the full dataset. Refer to details in Appendix H (Data Split Methods).

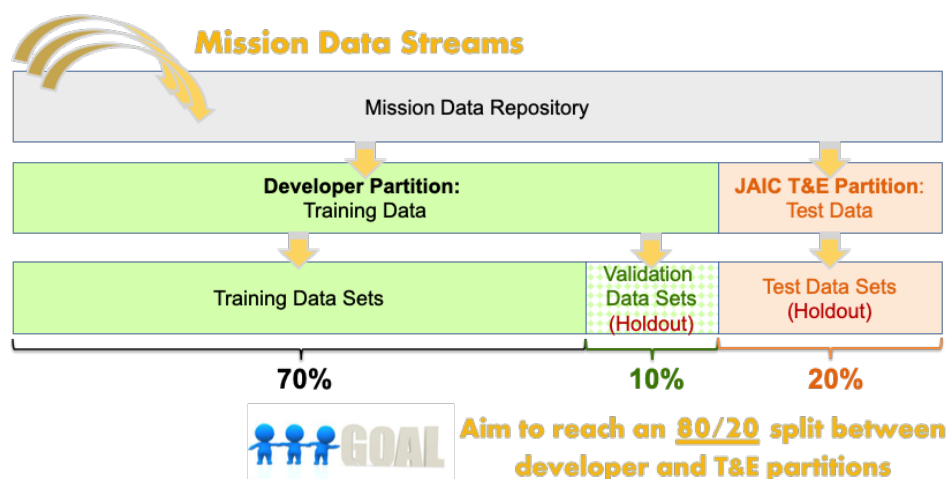


Figure 6-3. Data Splitting Schema

The ability for testing to yield statistically significant results is critical yet can be difficult for rare event or low probability classes as well as when human testing is involved. Here we discuss challenges related to the data itself. In use cases where the prediction of rare events or low-probability classes are of high operational importance, testers may intentionally opt to

oversample these instances as a means of countering this class imbalance. With oversampling, instances from the minority class(es) are reproduced and reinserted into the training split to overcome observational skewedness. Models that implement stochastic gradient descent are particularly sensitive to oversampling due to the iterative optimization of the loss function. Oversampling may also result in significant overfitting. If the original class imbalance is extremely skewed, learning time may also increase significantly when oversampling is applied.

Conversely, testers may instead adopt an undersampling strategy, in which instances belonging to the majority class(es) are dropped from the training set to level the distributional ratio between classes. Undersampling can often produce undesirable model fits, however, since a significant proportion of instances are randomly removed from training, the model learns a limited set of features from the majority class that do not accurately represent the true feature distribution of its members. If necessary, testers should therefore consider employing a combination of oversampling and undersampling to mitigate the risks posed by these individual strategies. Alternatively, metrics can be targeted for specific classes or outcomes. Crucially, both oversampling and undersampling should only be applied to the training set while the validation and test sets should reflect a stratified class distribution observed from the overall dataset before splitting; this approach increases the likelihood that the evaluation will reflect the natural data.

In cases where the full dataset contains only a small number of instances, testers may also opt for a k-fold cross-validation strategy described in Figure 6-4. Unlike train/validation/test splits where each split varies in size, the cross-validation method produces k groups of equal size without replacement. For each iteration of the training cycle, different single groups are used for testing while the remaining k-1 groups are used in each training phase; therefore, instead of directly obtaining performance metrics from a singular test set as with conventional data splits, we instead average the results of k train-test trials. K-fold cross-validation may also be preferred when there is a highly unbalanced dataset, such that most instances belong to a small number of classes and a small number or proportion of instances belong to several independent classes. Even k-fold cross-validation can result in unanticipated errors in operational use, so a hold-out set is preferred.

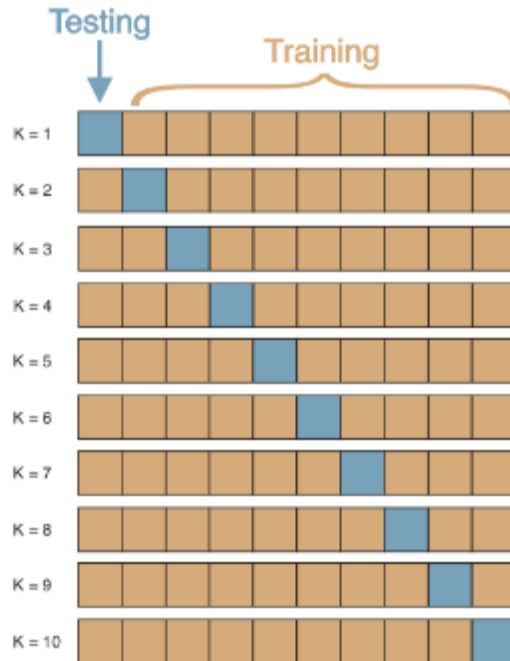


Figure 6-4. Algorithm for 10-Fold Cross Validation

While not as desirable as operational data, synthetic training data may be generated in situations where the amount of available domain data is low or where the expected event is a rare event problem. A future data environment must provide tools to enable the generation of synthetic test data that reflects operational data sufficiently to give meaningful and accurate results while reducing the data collection and annotation burden . Synthetic data development should aim for a repeatable process to support future AI projects. Figure 6-5 below depicts the high-level stages and processes necessary to produce synthetic data.

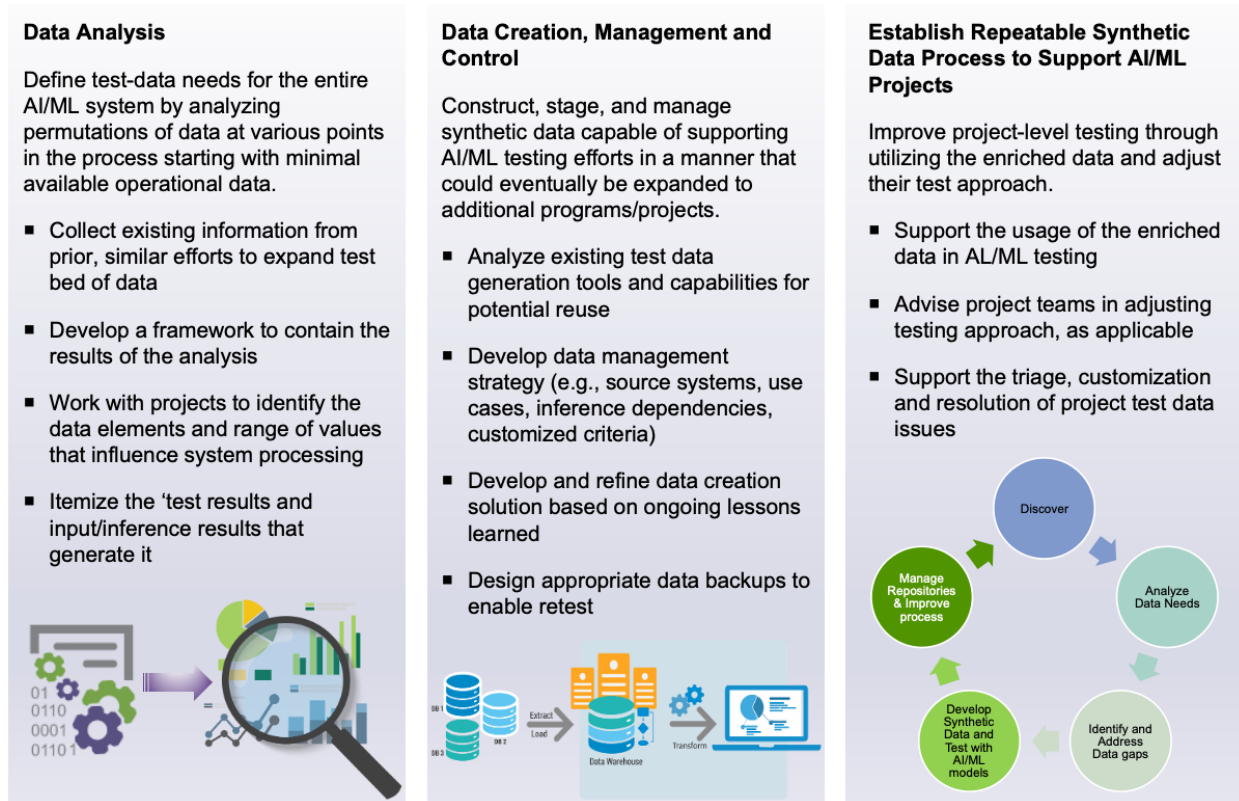


Figure 6-5. Synthetic Data Approaches to Achieve Repeatability

During neural network training and retraining, developers and evaluators should consider the effect of batch size, which is the constant increment of unique data instances fed to the model in a given subsample and then repeated for all other subsamples until the model has observed the entire dataset. For example, for a dataset containing six million instances, a batch size of 32 will result in 187,500 batches. After each batch is processed, neural models will evaluate their predictions against the “ground truth” reference values of the dataset and update the model parameters learned in the previous batch to minimize this error rate via gradient descent. Because the gradient descent algorithm estimates parameter adjustments based on the average error across batch instances, batch size can often impact the quality of each model update. Given that smaller batch sizes are statistically more likely to produce greater variation than batches containing larger samples, each iteration may introduce noisy fluctuations with each parameter update. Conversely, larger batch sizes allow the model to observe a more representative distribution of examples in between updates. The strength of each update is further modulated by the model’s learning rate, or the coefficient by which weights are readjusted based on new observations in each batch; higher learning rates generally result in more significant parameter updates while lower learning rates produce modest incremental changes. Depending on the complexity and variation of the data, small batch sizes and high learning rates may produce unstable models that fail to converge on an optimal fit. Previous studies on the correlation between batch size and model fitting recommend an initial batch size of 32 and a default learning rate of 0.01, although developers are strongly encouraged to monitor fluctuations in the model’s error rate during training. Based on their findings, developers can then experiment with different learning rates

and batch sizes until performance on the training and test splits approximate one another and iterations stabilize to produce minimal improvements.

6.1.2.5 Cybersecurity Considerations During Development

The “learning” nature of most AIES (those which rely on ML) leads to AI models whose behavior may not be transparent or explainable. As a result, cybersecurity considerations must include a clear understanding and protection of the AI development process. Historically, AIES have not been designed with cybersecurity threats in mind. Only in the past decade has there been a growing awareness of the nature of such threats. Threats may be introduced in many places during the MLOps development pipeline, and testing and evaluation may not reveal system compromise except under very specific, potentially unknown circumstances.

Modern AIES development has been greatly enabled by the use of widely available datasets, common open-source libraries, and prebuilt classification/regression models. Each of these constitutes a threat vector, as they provide a link from the untrusted creators/contributors to the AIES development process.

As the number of publicly available datasets increases, developers are increasingly training ML systems on data they did not create or do not control. This uncontrolled data is a threat vector, as a variety of attacks involve introducing adversarial data into the datasets—called data poisoning—in order to influence the AIES behavior. An ML system that learns from poisoned data may output random simply incorrect, or deliberately malicious values according to the attacker’s plan. Tracking data provenance, signing or certifying data, and controlling dataset creation and storage can protect against data poisoning attacks in the training stage.

Protecting the AIES development process also requires tracking and ensuring the security of the libraries and dependencies that define the AIES. The supply chain for an AIES application may involve software dependencies from local libraries, open source, or third-party sources. In parallel to the growth of dataset numbers, the number of software libraries is growing, and these tools are finding their way into most, if not all, major AIESs. Some major packages are the Python tools PyTorch, Tensorflow, and SciKit-Learn, which provide a solid framework for model development. These software packages may also represent a threat vector, as adversarial alterations could introduce unexpected behavior into the models they create. Developers must be aware of software provenance and be vigilant about integrating publicly available libraries into the AIES.

In addition to prewritten software packages, prebuilt ML models are also increasingly available. These models may be integrated as-is into an AIES or may be used as a general foundational model, which is then built upon to create a task specific model (transfer learning). Again, they represent a threat vector where an adversary can introduce desired adversarial behavior into the model before it is integrated into the AIES. Developers must be aware of model provenance and be vigilant about integrating publicly available pretrained models into the AIES.

A summary of known attacks on the ML pipeline can be found in 7.4Appendix I.

6.1.2.6 AIES Ethics

Ethical AI principles are defined in many U.S. based and international forums. The DoD has principles that will “apply to both combat and non-combat functions and assist the U.S. military in upholding legal, ethical and policy commitments in the field of AI”.

“The department’s AI ethical principles encompass five major areas:

Responsible. DoD personnel will exercise appropriate levels of judgment and care, while remaining responsible for the development, deployment, and use of AI capabilities.

Equitable. The Department will take deliberate steps to minimize unintended bias in AI capabilities.

Traceable. The Department’s AI capabilities will be developed and deployed such that relevant personnel possess an appropriate understanding of the technology, development processes, and operational methods applicable to AI capabilities, including with transparent and auditable methodologies, data sources, and design procedure and documentation.

Reliable. The Department’s AI capabilities will have explicit, well-defined uses, and the safety, security, and effectiveness of such capabilities will be subject to testing and assurance within those defined uses across their entire lifecycles.

Governable. The Department will design and engineer AI capabilities to fulfill their intended functions while possessing the ability to detect and avoid unintended consequences, and the ability to disengage or deactivate deployed systems that demonstrate unintended behavior.”

Operationalizing these principles is an area of active research and discussion. While teams and established processes are being determined it is recommended that development and testing teams be able to provide qualitative responses to these principles and to include them in plans (e.g., requirements, operational use cases, test measure). These assessment plans and results would be shared with the vendor and evidence collected to support their application. These results will help to inform on risks and define mitigations that may be warranted, depending upon the mission and application of the AIES.

6.1.2.7 Performance Metrics for AI Models

This section highlights some basic performance metrics to enable the T&E of AI models (focus on ML). These metrics should be shaped by the type of AI model, the intended capability, and mission as defined in the Evaluation Card.

This is by no means an exhaustive list,¹ and the proper metrics depend on information need and the type of AI. For instance, for applications such as information retrieval or recommender systems, have metrics based on answer ranking, such as mean average precision. Metrics can fall into two major areas: (1) metrics to assess the performance of the model itself and (2) metrics that inform of the overall process used to create and support the AIES. All of these are useful to inform the overall performance of the system (effective, suitable, survivable), yet insufficient to quantify overall effectiveness of the system. These metrics can be measured through commercial tools and shared with testers to inform performance. Standard AI model training tools can collect these measures, which provide the ability to trace to the model and the training data.

¹ Measures are the types of things that a metric will capture. For instance, algorithmic performance on a classification task uses precision/recall. Other measurables include assurance factors such as robustness (e.g., ,), which could have metrics such as P/R or other types.

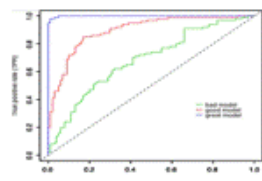
Standard measures exist for evaluating and training AI models, but they vary depending on the type of AI (e.g., classifier, regression). A classifier is an AI model used to organize or map data to discrete classes (e.g., car, truck, motorcycle). Typically, a frontline, but not only possible, measurement is precision/recall for binary classifiers (e.g., a thing is in a class or not). Precision captures the percentage of results reported as correct that are correct. Recall captures the percentage of correct results that are reported as correct. Sometimes these are combined, looking at both correct results and incorrect ones using predicted and actual positives and negatives as specified in Table 6-2.

Table 6-2. Confusion Matrix

		Actual/Measured	
		Positive	Negative
Predicted	Positive	TP (True Positive)	FP (False Positive)
	Negative	FN (False Negative)	TN (True Negative)

Positive and negatives can be further refined to look at performance over subclasses or smaller classes (e.g., vehicle can be decomposed to car, truck, van, tank). From these measured values, one can calculate several standardized measures of performance for classifier models, as shown in Table 6-3.

Table 6-3. Performance Measures for Classifier Models

Precision	Corresponds to how many classifications are true positives	$\frac{TP}{TP + FP}$
Recall	Corresponds to how many true positives were found	$\frac{TP}{TP + FN}$
Accuracy	Percentage of correctly classified items over all items	$\frac{TP + TN}{TP + TN + FP + FN}$
F1	A weighted average of precision and recall	$\frac{TP}{TP + \frac{1}{2}(FP + FN)}$
Receiver operating characteristic (ROC)	Graphing of a false positive rate versus the true positive rate (noting the curve shape versus the area under the curve)	

Regression models deal with a continuum of data, e.g., what is the predicted grade based upon number of hours studied. Some measures defined for regression models are shown in Table 6-4.

Table 6-4. Performance Measures for Regression Models

Regression performance	Mean square area (MSE) used for measuring error often used for training	$\sum_{i=1}^N \frac{(x_i - \hat{x}_i)^2}{N}$ $\hat{x}_i$ = estimated x_i = actual observed N = number of data points
	Mean absolute error (MAE) provides intuitive linear error reporting.	$\frac{\sum_{i=1}^n y_i - x_i }{n}$ y_i = prediction x_i = true value n = number of data points
	Root Means Squared Area (RMSE) used to evaluate and report performance.	$\sqrt{\sum_{i=1}^N \frac{(x_i - \hat{x}_i)^2}{N}}$ $\hat{x}_i$ = estimated x_i = actual observed N = number of data points

Unsupervised learning model performance can use measures to examine portions of data within a dataset. The following are some examples, but the most common application is to compare the unsupervised training results to supervised learning performance [4]; however, given the typical use of unsupervised ML as an exploratory task, these are rarely applied.

Reinforcement learning models are trained using unlabeled data as well as a reward system to enable the model to learn optimal solutions. Their T&E, as discussed in Section 7, can be more complex.

- Sum of discounted rewards/sum of rewards is the most common measure.
- Time to achieve desired performance is also used.

Deep learning models are trained using large amounts of unlabeled data in artificial neural networks. Typical measurements include those related to training and network structure such as programmability, latency, accuracy, throughput, energy efficiency, and rate of learning [5].

6.1.2.8 Reproducibility

The field of AI is often criticized because the results of AIES are not reproducible. Gundersen & Kjensmo define reproducibility as “...in empirical AI research is the ability of an independent research team to produce the same results using the same AI method based on the documentation made by the original research team.” This can be applied to the systems engineering of AIES because the reliability of a system can be questioned if the AI-enabled functionality does not perform consistently or effectively. The likelihood that a system will be successfully implemented and used in a production environment can depend on system functionality being reproducible. Managing data pipelines is necessary to measure reproducibility. Refer to the Section 6 of this report for additional information on data management.

At the core of software development is the ability to manage source files when commercial tools like Git, Mercurial, or Perforce products are used. Source control provides the ability to trace errors and make fixes, leading to a repeatable software development process. Teams also benefit on source control to support engineering collaboration. The ability to trace changes downstream

allows team members to understand the current release or to source a new issue or error with the use of branches and source control.

Reproducibility for an ML model is essential for an engineered system. Engineers need to be able to recreate results and build pipelines to the model pipeline. Storing raw data does not provide cover for the story of what was added to the model. Provenance, or history of data, is key to understanding the behavior of a model. Data may come in any form: structured or unstructured data such as images, video, or audio. Data cannot be used in these raw forms and must be transformed for use as a feature for the algorithm to use it. Features are data properties and are used to make predictions.

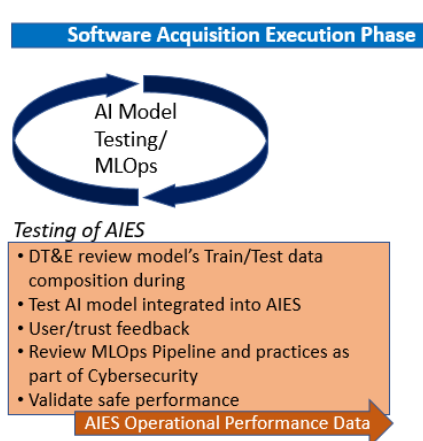
Data pipelines are the tools for transforming data where they occur in training and during inference. When the data model is trained, it is trained to make predictions. Inference is when the model makes a prediction. During training, it is important to track data used to build the model. Data tracking is useful to understand distributions of the original data featured and critical for audit requirements. Incorrect formats may be as broad as providing a black and white model when a color image is expected. As more companies start deploying multiple ML pipelines that share common features, wider adoption of feature stores will occur. Feature stores use “store” as a place to keep transformations and feature extractions used across multiple ML applications.

Different artifacts are used among team members in modern software development. Sharing of a range of material is essential to expand ML success. Training is important to capture the hyperparameters that are used. Without these starting points, it will be difficult to add new team members to progress with model improvement. While objectives may change in business, ML systems are more sensitive to changes from data and the quality of prediction can degrade from new data. That is the reason training data may need to be refreshed often. Systems and controls need to be in place to ensure production data is sampled and captured for retraining.

Data can grow large in volume and in capacity. For example, satellite or medical imaging can easily stretch into gigabits in size per file with million to billions of rows per day. This a major challenge.

There is a rich history of ETL tools used to move and transform data. While it can be used in ML, new tools are also appearing. Tools are spanning a wide range of abilities. Also, tracking experiments and new training tools are also coming out, such as SaaS Toolkit. Weights and Biases and ClearML are two new tools for tracking experiments over time.

6.2 T&E of AIES



This section discusses the T&E execution of the AIES focusing on the following topics:

- Model Test & Evaluation
- Cybersecurity Test and Evaluation of AIES
- Operational Test & Evaluation of AIES
- Measure of Users Trust

6.2.1 Model Test & Evaluation

Evaluators should agree on a common T&E framework in which to uniformly record their observations from dataset profiles, model training runs, system-generated logs, and user-oriented requirements and recommendations. Ideally, all documentation and artifacts should reside in a single repository or within a few tracking platforms and tools as possible to avoid collaborative risks like knowledge silos, whereby a subset of the T&E team record their findings without the awareness of other members. For example, while subversion repositories are often used for tracking code updates, bug reports, and usability issues, team-oriented knowledge management tools like Confluence and SharePoint are frequently preferred for storing data products and documents. As described in prior sections, there are strong motivations for developing consistent formats for semi-structured templates that identify requirements, specifications, results, and user-based experiences.

Most ML libraries are prepackaged with model evaluation utilities for conducting statistical analyses of the model's results and obtaining common performance metrics like precision, recall, accuracy, and F_1 scores from their prediction outputs. Such utilities can facilitate the evaluation of model performance, especially when different members of the T&E team are assigned to different ML software products. However, these products may differ significantly in terms of the metrics they can compute "out of the box." For example, many libraries do not give the option for the user to define their own F-score (e.g., $F_{0.5}$ or F_2) depending on their unique task-based or operational thresholds, and some may include other useful performance metrics like Receiver Operating Characteristic and Area Under the Curve, while many do not. Similarly, some utilities may offer different variants of a family of performance metrics. For example, pointwise mutual information (PMI) is commonly used in Natural Language Processing (NLP) classification tasks to compute the distributional dependence between terms in a dataset and may produce different scores based on the particular variant implemented, which can range from positive PMI, normalized PMI, and logarithmic PMI-k. Some utilities (and some users) may conflate these different variants or fail to note their mathematical differences when comparing scores. As a matter of good practice, it is therefore advisable for evaluators to choose a standalone statistical package or single utility with which to assess the performance of all models to avoid such risks.

Evaluators should also document the specific selected implementation of performance metrics in the Evaluation Card.

Beyond traditional performance metrics, many ML use cases require some degree of predictive transparency so that users may assess whether the model's learning objective is informed by a relevant set of features and dependencies employed in human sensemaking. Many high-risk use cases, in which false positive and false negative thresholds are particularly low, demand some level of human-in-the-loop expert review to avoid potentially catastrophic consequences. Virtually all modern AIES rely on deep neural networks to learn input-output mappings by producing decision boundaries in high-dimension feature spaces while minimizing loss; in simpler terms, deep learning models observe complex patterns of features between instances and their targets during the training phase while adjusting the weights of patterns learned in between training iterations to reduce model error. Because deep learning models are free to optimize their own internal representations without direct supervision from human experts, a trained model will often learn to prioritize some goal without sensitivity to rare edge cases or may otherwise fail to discriminate features that experts consider critical.

6.2.2 Cybersecurity Testing for AIES

DoD acquisition programs use National Security Agency trained Red Teams to engage and conduct Cybersecurity T&E assessments to report on a system's ability to withstand, deter, or defend themselves from cybersecurity threats, including those to AIES. These test activities will leverage threat documentation defined in the early stages of acquisition (e.g., VOLT) that includes specific considerations for the planned AIES. It is important to ensure the DoD leaders who are planning and executing these assessments are trained on the benefits, risks and threats associated with AI (e.g., Defense Acquisition University's (DAU) Artificial Intelligence Intermediate Credential [CENG 003] which is currently in development).

Traditional methods for cybersecurity testing include code scans and employing commercial test tools, as well as the conduct of more formal testing events such as Cooperative Vulnerability Identification (CVIs), Adversarial Cybersecurity DT&E (ACDs), Adversarial Assessments (AAs) and Cooperative Vulnerability and Penetration Assessments (CVPAs).

CVI: identifies known vulnerabilities in HW/SW, interfaces, operations and architecture in order to assess the mission risk associated with those vulnerabilities and to determine mitigations or countermeasures.

ACD: includes evaluations of a system's cyber survivability and operational resilience in a mission context, using realistic threat exploitation techniques, while in a representative operating env

AA: assesses the ability of a unit equipped with a system to support its missions while withstanding validated and representative cyber threat activity.

CVPA: purpose of this phase is to provide a comprehensive characterization of the cybersecurity status of a system in a fully operational context, and to substitute for reconnaissance activities in support of adversarial testing when necessary .

These methods may need to evolve to ensure the nature of AIES is fully accounted for. For example, AIES are trained on set of labeled data that teaches the system how to perform. One of the known risks is that the training data could be poisoned (e.g., include intentionally mislabeled

training data). This will cause the system to behave incorrectly and as sought by an adversary (e.g., recognize a tank as a school bus). To mitigate this, the cybersecurity assessment team may want to review the composition of the training data to understand the risk of this occurring (was the data from an open source). Looking at the practices followed to build the AIES informs on the risks of the final integrated system.

In addition, the following methods are recommended for the Cybersecurity T&E Leads.

Treat the model itself as a sub system that warrants assessment in addition to the overall AIES.

Use the system requirements and the operational use cases (defined in earlier stages) to ensure positive and negative scenarios are fully addressed (e.g., what the AI should/should not do relevant the role and skills of the user).

Review published case studies (such as those in Advanced Threat Landscape for Artificial-Intelligence Systems [ATLAS]) to inform the relevant use cases to assess the model and the AIES. Include any relevant adversarial threats into test data that either the model or the AIES is challenged to operate with.

Request and review the development team's practices to determine the protections in place (especially in contractor spaces) to protect the AI training data (data, model development and training environment, hyperparameters, and training logs) from any inappropriate actors who may use this early environment to introduce or access data used to later compromise the system once put into operation. (Altered training data, model probing for useful information about the model, integrity attacks causing specific performance errors, enabling the reveal of secure/sensitive data, availability).

Develop and employ Advanced Threat Landscape for Artificial-Intelligence (M&S) that can interact with the AI model and the AIES system fully explore ranges of performance to ensure edges where rapid degradation occur are understood.

Assess users' performance for the likelihood of inappropriately trusting the AIES when it should not be used (e.g., potentially been compromised). Assess how well the operator can detect, mitigate, or prevent these from happening.

Determine the system's ability to employ countermeasures to thwart attacks, self-manage, and sustain its performance while under threat. These threats should be informed by threat documentation developed in the early stages of the AIES project (e.g., VOLT). Does the system fail to operate, continue to perform, or start causing harmful/incorrect results while enduring the threat. This should include an ML models' ability to log its performance, users' responses to AI (agree/disagree), and the training data changes once fielded (drift). This will all be used to assess harmful impacts caused by AI and help with any troubleshooting performed since the system will likely need to be retrained and redeployed safely in the future. Anti-tamper safeguards, such as zeroization mechanisms and electromagnetic frequency shielding, should also be considered and rigorously tested if adopted to protect edge devices and embedded systems under adversarial field conditions .

Develop and employ automated adversarial testing to exam automated hardware and software updates to the AIES.

Government teams should ensure that additional resources, contract deliverables, and SME that are necessary to address these cybersecurity assessments are fully accounted for in the overall Evaluation Card or advise (based upon the threats) of the risks if they are not addressed.

6.2.3 Operational Test & Evaluation

The goal of DoD Operational Test and Evaluation is to determine if a system is operationally effective, suitable, and survivable . This includes:

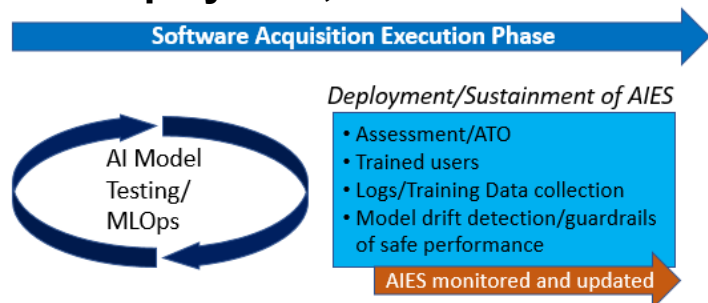
- Operational Effectiveness: A measure of the overall ability of a system to accomplish a mission when used by representative personnel in the environment planned or expected for operational employment of the system considering organization, doctrine, tactics, supportability, survivability, vulnerability, and threat.
- Operational Suitability: The degree to which a system can be placed and sustained satisfactorily in-field use with consideration being given to availability, compatibility, transportability, interoperability, reliability, wartime usage rates, maintainability, safety, human factors, habitability, manpower, logistics supportability, natural environmental effects and impacts, documentation, and training requirements.
- Survivability: Includes the elements of susceptibility, vulnerability, and recoverability. A survivability assessment should be conducted for all systems under OT&E oversight that may be exposed to threat weapons in a combat environment or to combat-induced conditions that may degrade capabilities, whether or not the program is designated for Live-Fire Test and Evaluation oversight . This also includes cybersecurity testing.

Recent DoD acquisition guidance (adaptive acquisition framework) encourages shifting operational testing left in the software lifecycle to collect operationally relevant test data earlier in the overall process. Additionally, testing in the operational environment may reveal system issues that can be addressed earlier in system development and reduce the risk of costly system changes. This is especially true for AIES, which have unique aspects of their performance that need to be assessed with controlled live testing before operational testing. The use of HMT, and directly involving users, in operational testing can foster improved system usability, performance, and adaptability to real-world scenarios. It is important to consider that AIES T&E includes both evaluating and documenting model development and a strict attention to detail during integration to catch potential errors introduced by the AI component, which may compound in integration. Operational data used throughout the development and integration phases will facilitate a smoother and more accurate operational test. During this stage, it is crucial to evaluate the results generated by the AIES. This allows for the further development and validation of operational use cases. Additionally, the performance of the AI model should be assessed to ensure it contributes to overall system performance and the establishment of operational scenario metrics. For an AIES it is especially important to evaluate additional sources of evidence in addition to the model selection and training, data selection, and labeling. These activities enable a more robust assessment and mitigate what may be cost prohibitive to explore in fuller OT. There is a total body of evidence that should be gathered across the lifecycle, enabled, and informed by the PMO and the contractor.

User trust feedback is essential in AIES development cycles and one of the key aspects to be collected and evaluated during OT&E. It is driven by factors such as requirements, operational use cases, involved user groups, existing measures, and relevant trainings after deployment. User feedback should be designed based on system requirement and serves as a vehicle to provide self-auditing on a periodic basis. Emerging practices are forming to define assessment, in a simple and repeatable manner, of a user's trust in an AIES. Refer to a Machine Learning Trust

Score (MLTS) survey and an explanation satisfaction checklist for measuring users' acceptance of model fit on the basis of AI explainability detailed in 7.4 Appendix K.

6.3 Deployment, Sustainment and Performance Monitoring



AIES deployment is the introduction of a new release into a production environment. Deployment includes training and business systems operations activities so that production users are fully supported. AI Model deployment is the process of enabling AI functionality within a production system. This section presents considerations when deploying AI functionality.

As use of an AIES grows, it may incur an expanded user base, deployment in a new operating environment, and be applied with additional functionality. These considerations should be considered a new fielding of the system and warrant an approach like the original AIES deployment. This would be different from an adjustment to an AIES where the model is retrained on a more limited basis with new data.

DoD Instruction 5000.2 states: “Sustainment includes supply, maintenance, transportation, sustaining engineering, data management, configuration management, manpower, personnel, training, habitability, survivability, environment, safety (including explosives safety), occupational health, protection of critical program information, anti-tamper provisions, and IT, including National Security Systems, supportability and interoperability functions.” Carnegie Mellon University Software Engineering Institute provides the following definition for software sustainment: “the processes, procedures, people, materiel, and information required to support, maintain, and operate the software aspects of a system” .

The following topics are covered:

- Authorization to Operate
- Model Deployment & Integration
- Continuous T&E, Model Monitoring and Maintenance
- Program Execution

6.3.1 Assessment and Authorization to enable the Authorization to Operate

All DoD Program managers are required to follow National Institute of Standards and Technology (NIST) Risk Management Framework (RMF) Special Publication 800-37 as the standard for Assessment and Authorization process. This process enables a program to receive an Authorization to Operate (ATO) by an Authorizing Official (AO). Recent efforts are underway to establish a greater emphasis on the continuous monitoring step of RMF to allow for continuous authorization (cATO) .

NIST is currently working to address AI within the RMF and in the meanwhile it has not specifically defined the implications of the presence of AI to the relevant controls that should be considered. Even without direct guidance, caution is urged to ensure that the RMF process is informed by many of the risks to the AIES. The framework and the controls are broad enough (spanning from policy, literacy, logging, acquisition processes) and can be used to inform on the general software risks of an AIES, perhaps in a similar way to any software system. The literacy of the security professionals on AI and its vulnerabilities and risks may enable them to lean forward and take a rigorous approach to RMF, perhaps considering some of the recommendations of this document. Notwithstanding, the RMF controls themselves may continue to refine to include additional guidance in the future to provide even better clarity and consistency of execution of the processes while accounting for the risks for each type and application of AI.

The implications of an AIES proceeding through a cATO process are much less well-defined. Caution should be taken when taking this approach as there likely needs to be more maturation before we can safely include these systems in this process. Attention should be paid to studying this as there is a great opportunity to marry the need for model retraining and redeployment with a more agile process that include continuous monitoring.

6.3.2 AIES Fielding and Integration

Once AIES models have met performance requirements under the development T&E phase and all ATO requirements have been satisfied, these systems are then transitioned into a production environment for deployment and integration. The operational requirements of the AI-enabled use cases will ultimately determine how the model shall be fielded by end users in different support roles: For example, remote sensing applications may require AI models to be strategically adapted and integrated locally on embedded systems or their peripheries to support low latency conditions and real-time field operations. Under a federated deployment framework, various users with different privileges and support roles may perform “learn-on-the-fly” updates on a subset of interdependent models given their operational need without having to interact with other components of the ecosystem. Decentralized or distributed deployment architectures enable various user groups, such as field operators, intelligence analysts, and command personnel, to receive critical AI support on data fusion tasks while streamlining data traffic and facilitating load balancing across computing resources. AIES integration decisions should therefore optimize the delivery of services while accounting and factoring for multiple operational trade-offs.

6.3.2.1 Cloud-Based and On-Premises Deployments

Cloud environments generally provide the greatest amount of high-performance computing power compared to other AI deployment options. Distributed clusters can host a large number of GPUs and tensor processing units (TPUs) to support massive parallel processing tasks in various AI applications, including computer vision/image recognition, NLP, and agent-based modeling or course-of-action generation tasks. Cloud-based deployments also enable greater scalability in terms of resource allocations compared to on-premises servers or edge computing deployments, since additional GPU and TPU nodes can be re-dedicated to the model at run time to meet high volumes of data, whereas local systems are inherently constrained by static configurations.

Commercial providers including Amazon Web Services, Google, Microsoft, Dell, Oracle, and IBM offer various infrastructure-as-a-service solutions, which greatly reduce the DoD’s level of

effort in integrating AI models compared to in-sourced cloud environments. Conversely, some previous DoD-oriented assessments have also advocated the benefits of securely integrating open-source cloud platforms like OpenStack and Cloud Foundry over commercially developed platforms when appropriate; such benefits include reduced costs and greater control and transparency in terms of performance monitoring. Cloud computing environments may fail to provide reliable access to AI services under degraded, denied, intermittent, and low-bandwidth operational conditions. For example, ground personnel may face communication interruptions and significant processing delays when attempting to remotely upload field data to a cloud-hosted model. Enterprise cloud environments also expose the system to an extended range of adversarial attacks and potential cybersecurity risks that would otherwise not jeopardize on-premises server integrations.

On-premises integrations generally provide less virtualization and scalability compared to conventional cloud-based solutions but can often deliver higher rates of data traffic across local networks. On-premises integrations also allow for stricter cybersecurity and physical security requirements, which can be directly managed in compliance with DoD policies. On-premises integrations also require considerable capacity planning in advance of model deployment since all infrastructure will typically reside within a single data center.

Regardless of the approach of deployment method, it is essential to implement automated adversarial testing as part of the deployment process to keep pace with automated updates to the AIES.

6.3.2.2 Edge Deployments and Hybrid Integrations

Hybrid deployment frameworks can leverage the benefits of cloud-based, on-premises, and edge computing while simultaneously minimizing the shortcomings of each individual deployment environment. Hybrid integrations are logically suitable solutions for federated AI approaches in which different distributed system components perform separate modeling subtasks that are subsequently aggregated and analyzed by a higher-order meta-model . For example, military personnel in the field may collect, annotate, and analyze disparate sources before they are shared with higher command staff and intelligence analysts . Such ground-level tasks are optimally supported by tactical edge subsystems that can be repositioned and interacted with as mission conditions and objectives permit. Autonomous systems may similarly depend on tactical cloud nodes to reduce latency and facilitate rapid AI support . Data observations and interim model states are then uploaded to more robust computing resources hosted on a remote cloud-based environment for deeper analysis and coordination. Federated AI applications can also expedite adaptive model refinements under domain drift conditions so that, instead of relying on a single model state at any given point in time, various independent model updates may be performed under a decentralized deployment architecture. Federated deployments, and federated edge learning architectures in particular, are resilient under conditions in which data collection and network communications are highly degraded . In federated edge learning, each device can continue to refine its own internal model based on its local data, and these submodel states may then be transmitted to central server where they are aggregated into a global model, which can then in turn be pushed to edge devices for further refinement . The combination of hybrid integrations and federated AI can also largely minimize what is commonly referred to as the multi-arm bandit scenario. Heterogenous or conflicting data streams and numerous incoming client requests can often destabilize a single model's state and produce traffic bottlenecks

between users and the model repository. Hybrid integrations enable users to offload data and model updates from edge devices to a common cloud service in batches for aggregation, fusion, confidence estimation, and higher-order inferencing.

6.3.3 Continuous T&E, Model Monitoring and Maintenance

The adoption of well-defined protocols for managing source code versions and pipeline interdependencies is essential for maintaining any system development task but is especially critical for AIES, as specified under the MLOps framework. In conventional software development, developers will create a new branch in a subversion control system (e.g., Git) whenever the codebase is updated. Under AIES development, however, many codebase updates will often introduce “knock-on effects,” which create a cascade of additional downstream or (more rarely) upstream integration issues between AI-adjacent pipeline components. These dependency-specific issues must then be identified and resolved before system integration can occur. Integration testing criteria should be initially defined from the project’s outset and should be continuously reformulated as unforeseen implementation issues and fixes emerge to ensure that each configuration modification is successfully scoped, documented, and integrated.

Conversely, other components of an AIES pipeline may be updated after the initial deployment to accommodate evolving data sources and data streams; for example, ETL applications, data preprocessors, and filtering modules may be integrated later in the lifecycle to replace or complement existing components from an earlier deployment . These updates can often introduce cascading system integration errors or new security vulnerabilities that may critically impact or halt the production environment . Given the rapid advancement of machine learning and deep learning technology and their applications in AI, newly released versions of open-source and commercial packages may differ substantially from older versions previously integrated into an AIES. Such upgrades often require other library-specific dependencies to be updated and installed within the pipeline environment. Conversely, if custom AI-support pipeline components are developed “in-house” by the DoD or through contracts with private subcontractors, the DoD team can provide further technical specifications to help shape additional refinements and define the testing effort required prior to reintegration.

Model performance will often degrade over extended periods because of domain drift when significant proportions of operational data streams are categorically different from the data sources originally used for model training prior to deployment. Within-domain distributional shifts may also occur when the relative frequencies of each data class change considerably over time. When model performance is degraded, the approach to resolving the issue may depend on the specific nature of the degradation and the operational requirements of the AIES. Degraded performance can be caused by a variety of issues, such as changes in the underlying data distribution, issues with the local model updates, or problems with the aggregation process . As deployed models are provided with production data that differs significantly from the data on which they were trained, AIES pipelines will often reach the edge of their current performance envelope. Several continuous monitoring techniques, such as Mahalanobis distance-based confidence scoring and out-of-distribution detection for neural networks, enable AIES production pipelines to detect unseen or uncharacteristic inputs and flag model states by providing quantified measures of predictive uncertainty. Additional data perturbation techniques that generate and inject user-defined levels of noise can often help T&E teams characterize the

source of model overfitting. Such data-driven evaluation methods are instrumental in empowering model sustainment efforts over the course of AIES delivery.

Automated testing and EDA routines must be implemented to assess model performance in near real-time and monitor the emergence of covariate shifts, prediction anomalies, and drops in confidence estimations . Any updates to the global model must be thoroughly tested before they are pushed and committed to edge devices . To help identify the source of model degradation, operational T&E teams can leverage methods to measure feature importance and apply prediction interpretability methods such as local interpretable model-agnostic explanations and Shapley additive explanations . Depending on the nature of the degradation, it may be necessary to re-train the model, adjust its hyperparameters, or even revise the overall AI model architecture. In federated or edge deployments, additional weight may be given to local models from devices that have processed higher-quality or more representative data as determined by the operational T&E team. Secure aggregation methods must also be followed to ensure that each local model's update is accurately incorporated into the global model . It is also critical that all previous model states are securely backed up and well documented to facilitate potential rollbacks in case future updates demonstrate poor performance once deployed.

Lastly, the relevant teams entrusted with sustainable deployment and continuous integration must collaborate to develop agile methodologies for eliciting and capturing user experiences and feedback once an AIES enters a production environment. Feedback instruments must allow users to provide both structured feedback (e.g., ratings, multiple-choice questions) and unstructured feedback (e.g. free-text comments) to afford users the freedom to document both prevalent technical issues and their own personal experiences for quantitative and qualitative analysis. Sustainment teams are encouraged to implement a user experience reporting tool that is specifically suited to the user's needs and the AIES engineering requirements. Such a tool should give users, developers, testers, and integrators the appropriate level of visibility so that progress on service and sustainment tickets may be easily tracked. Additional capabilities that facilitate logging and activity monitoring on both system performance and user interaction fronts may be wrapped into the feedback support tool.

To ensure long-term sustainability, system integrators should therefore focus on the following continuous SELC tasks outlined in Table 6-5 below.

Table 6-5. Sustainment Areas and Continuous SELC Support Tasks

Sustainment Areas	Continuous SELC Support Tasks
Data Management	• Assure data provenance and restorable backups • Maintain relational links between data sources, model predictions, and associated metadata stores • Monitor compliance for all encrypted traffic, model encryption, and encryption-at-rest policies • Update new access policies to reflect changes in user roles and workflows
Domain Monitoring	• Automate data profiling to assess significant changes in distribution, sparsity, and redundancy

Sustainment Areas	Continuous SELC Support Tasks
	Periodically compute data divergence metrics (e.g. Kullback–Leibler/Jensen-Shannon/Kolmogorov-Smirnov divergence tests)
Log Management	- Ensure that all operations, processes, and user requests are adequately and automatically recorded - Reconcile/fuse logs generated by different ecosystem components
Algorithmic Sustainment	- Periodically perform regression tests and re-fit deployed models as required - Configure automated performance alert triggers to inform model re-training and re-testing schedules - Implement anomaly detection and reporting capabilities - Evaluate meta-learners/ensemble learning (e.g. voting, boosting) methods to update/correct static model deployments as appropriate - Leverage emerging advances in explainable AI (XAI) including toolkits
Hardware Optimization	- Actively monitor resource utilization (CPUs, GPUs, TPUs) and loads across nodes and clusters - Evaluate different accelerator configurations - Leverage design space exploration tools
User Training	- Develop tutorials and reference materials in accessible language while avoiding technical terminology whenever possible - Document questions and technical issues and update training references accordingly - Partner non-technical users with data scientists, developers, and other experts
Feedback Elicitation	- Capture and align user requirements with continuous integration/development tasks - Establish stakeholder engagement activities with analysts, developers, and program office personnel - Document operational barriers preventing broader AI adoption

7 Exemplar SEPTAR Use Cases

Use cases describe the mission context of a system and the requirements for its operation and use. The purpose of a use case is to shape requirements and the T&E of those requirements.

This section contains three types of use cases:

- Perception (Section 7.1)
- Sensemaking (Section 7.2)
- Course of Action Generation (Section 7.3)

A general overview of each use case is provided under each of the above subsections along with a list of their specific requirements for T&E tasks. The additional subsections that follow then provide a series of considerations for each phase of the systems engineering life cycle as they pertain to T&E, including contracting, development, and sustainable deployment. The use cases discussed in this section were selected based on the diversity of AIES support areas that they collectively represent but it is worth noting that they are by no means exhaustive and there are several other AIES use cases that may comprise various mission objectives. Future use cases will encompass complex AIES such as those required in Joint All-Domain Command and Control (JADC2)/Multi-Domain Operations (MDO) where the complexity can arise from a) multiple ML components; b) complex data sources or data interactions; c) degraded environments and/or edge requirements; and d) multi-system operations.

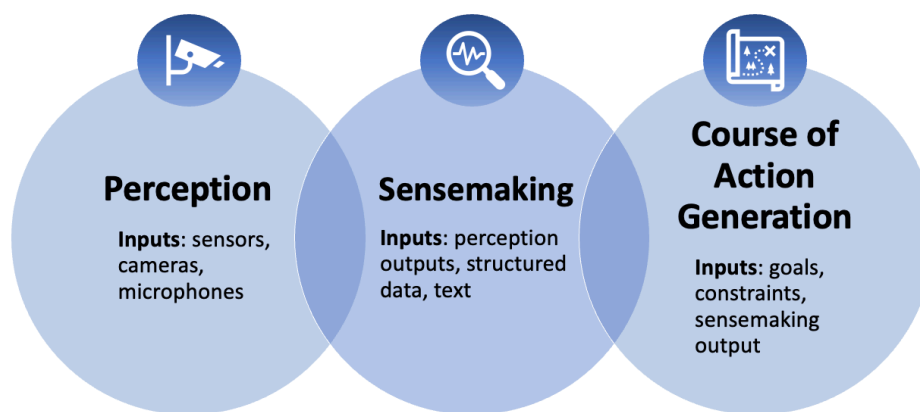


Figure 7-1. ML Use Cases

7.1 Perception (Use Case 1)

Perception covers a range of recognition or categorization tasks, such as automated speech recognition, computer vision for image recognition, and identification of objects/elements from sensor data. Since perception tasks, by and large, are categorization tasks; they have the ability for ground-truth evaluation, which is based on knowing the output the system *should* produce for a given input. Where they tend to have a greater T&E design burden is in the variation of the sensor or information capture tools, such as microphones or cameras. The use case is one of image recognition or identifying objects in images.

7.1.1 Use Case Description for Image Recognition Task

Description: Given a set of images taken from a Manufacturer X camera, with resolution of 1200 dots-per-inch (DPI), mounted on a tower at 50 feet at an angle of 45 degrees, detect adversary armored vehicles. The list of armored vehicles to be detected is provided. The images will be sent to the recognizer at a rate of one per minute. The system will send the analyst an alert giving the model of the vehicle recognized and time of recognition. Table 7-1 lists the specifications for this notional image recognition use case.

Table 7-1. Image Recognition Use Case Specifications

Specification Type	Dimension	Specification
Task-based Specifications	Report	Armored vehicle identification and response time for recognition.
	List of Classes of Armored Vehicles	Vehicle Type 1 Vehicle Type 2 Vehicle Type 3
	Time Marker in Zulu Format	Time between image taken and first result reported not greater than 15 seconds
		Throughput: 1 image per minute
	Camera specifications	1200 DPI
		Angle of image: 45 degrees from horizontal on tower mounted at 50 feet
		Distance from image target: less than 100 yards
	Environmental requirements	Percentage of cloud cover above which no recognition is possible 60%
		Images must achieve f-measure of 0.9 when images have buildings and sand-based ground
		Weather-related: recognition must work at 0.65 f-measure in rainy conditions
	Adversarial recognition	0.65 f-measure for deliberate disguising of tanks
	“Other” object categorization	0.8 f-measure for categorizing “other” vehicles – out of class
Workflow Measures	Time currently spent on task without AIES support	Analyst routinely monitors inputs from 2 cameras and each one takes 3 minutes if no vehicle is identified, 6 minutes if one is identified
	Task frequency	Once per minute per camera
	Number of people on task	Dependent on number of cameras
Operating Environment	Software interaction with other systems	Alert recording system
	Software interaction with user	Yes

Specification Type	Dimension	Specification
	Computational resources required for operation	Model must operate on current operational workflow equipment
	Precision/recall tradeoff	Analysts must be surveyed for precision/recall trade and system tuned for it

7.1.2 Related T&E Impacts for Each Phase of the SELC for Use Case 1

The general considerations that must be specified for image recognition relate to camera specification(s), categories of objects to be recognized, environmental factors that could interfere with successful image recognition, speed, and communication of results to users or downstream processes. The requirements analysis that follows illuminates the potential considerations for this use case as well as other perception ones. The important lessons learned are:

- Specification of the input providers, such as sensors, is critical to understand and test for operational impact.
- Specification of environmental factors, such as weather conditions or time of day, enables testers to test for important conditions that ensure operational success.
- Understanding the data is essential to account for potential class imbalances and applying this understanding to testing ensures that the AI components will “get the right things right”.
- Understanding the relative accuracy needs provides a system that matches the user’s needs and expectations.
- Acquiring, providing, conditioning, and annotating data has significant impact on cost and schedule. Tracking and retaining rights to data yields benefits.
- Post-deployment monitoring requires retention of data, component results, and uses provides the ability for continuous testing for detecting and mitigating degraded performance.

7.1.2.1 T&E Considerations in Requirements Definition for Use Case 1

Because perception tasks are heavily dependent on equipment such as sensors, microphones, and cameras, defining requirements that list equipment specifications and environmental factors that the AIES is expected to perform with is important. Different applications will require varied specifications. For instance, in voice recognition, background noise can interfere with successful recognition, and therefore reduce recognition rates.

For image recognition, the requirements must convey the camera specification(s). Limiting them to one camera may limit their flexibility and performance when deployed for new camera types. The most important elements are the DPI, the camera angles, and placement distance in height and length. In addition, environmental conditions such as weather and terrain should be specified. While operational test may be difficult with rarer weather events, requirements should delineate the typical weather conditions at a minimum. Should any of these specifications deviate

from what the underlying model was trained on, the AIES performance could degrade significantly.

When the image recognition involves class imbalances (i.e., important objects pass by the cameras infrequently compared to other object types), specifying the per-class recognition rates in terms of precision and recall positively impacts model effectiveness. If a recognition model conveys its output to another component, then error thresholds should reflect the dependencies of the downstream processes. While training is the most computationally expensive stage, requirements may need to reflect the computational needs of deployed models during inference.

Finally, recognition thresholds must be balanced. The two standard measures are precision, or the proportion of recognitions that are correct, and recall, or the proportion of correct recognitions out of the image collection. In many trained algorithms, one is traded for the other, and the emphasized one depends on operator preference, actions of the system based on recognition, and the criticality of the operation. If, as in this case, an operator is responsible for reports or actions based on images and the number of images is large, they may prefer increased precision to reduce the number of images they must verify or act on.

7.1.2.2 T&E Considerations in Contracts for Use Case 1

Collecting and curating sufficient training and testing data should be a focus in the contracting phase. The data must reflect operational data, which is often the biggest cost of building and evaluating systems. Collection is costly if current systems do not support the ability to capture and store data as a side effect of their operation, or if there are legal, ethical, or security concerns related to the data. Curation for some tasks requires the definition of an annotation standard, annotation of the data, and resolution for inter-annotator agreement. This can represent a significant cost for both developmental testing and more formal T&E due to the need for human annotation. An increasingly common approach is contract annotation and tools that can speed the annotation process through bootstrapping. Regardless of the method, the contracts should define expectations to ensure the data is available in sufficient quantity to ensure all aspects to include T&E.

Depending on the annotation strategy, image annotation can represent a significant proportion of schedule and monetary costs. For instance, some ML approaches require pixel-level annotation. By contrast, automated speech recognition requires transcriptions of roughly 10 hours of speech for a basic capability. For T&E of image recognition, more data is needed to have accurate annotation for testing. Contractually, the rights to models and annotations are vital to specify for the government to acquire and retain rights. In addition to the models and annotations, the government should ensure that the contract specifies rights to and ability to test data pre- and post-processing tools, such as image cleaning tools. Government rights to datasets, features, algorithms, configurations, design documentation, and trained models can both contribute to reproducibility and avoid vendor lock. While this practice can discourage smaller companies and startups, tailored rights agreements can mitigate the risk to these companies while allowing government flexibility. Finally, the contract should specify that the developer provide “hooks” into the ML-based components to facilitate continuous evaluation.

7.1.2.3 T&E Considerations in Development and Integration for Use Case 1

While perception tasks are natural fits for ML, training models depends on sufficient annotated data. From a development perspective, this means project plans must account for sufficient time

to acquire, clean, and annotate data, which can represent up to 90 percent of cost and schedule. It is particularly essential to ensure that enough operational data has been annotated to perform statistically significant model evaluations. Model development when a paucity of images or relevant images exists can be enhanced using pre-existing models or datasets, often from open-source sites. For instance, users may use open-source images of tanks to augment operational pictures of tanks. While these allow for more rapid development, developers must perform more testing for bias and adversarial AI prior to training and testing, and where possible, mitigate for both of these. This should include open-source models or data and schedules should reflect that need. Logging of training information is important, particularly when evaluation turns up unexpected errors. Model Cards should be prepared as well.

Evaluation data should always come from operational sources. Image recognition training and testing often relies on high performance computing resources, whereas the models should be able to operate on less computationally intensive platforms. Developers should also make meaningful confidence values available for the user or downstream modules. Pre-processing modules for image cleanup may be necessary and can take the form of changing the DPI or adjusting image clarity. These can be individually tested, although could require additional human resources. One approach to mitigate this is to degrade ideal images through software means.

Perception components tend to be data transformation and are often embedded in a larger workflow. NLP perception applications provide transformed data to another component in the system, such as optical character recognition (OCR) generating text from images. The integration may involve testing for characteristic errors and producing post-processing modules to reduce errors. For example, post-editing of OCR may involve reintroduction of diacritics (accent marks) into the text through specialized spell-checking. While in this use case, object recognition may be direct presentation to the user or it may be integrated into alert systems or report generation systems.

7.1.2.4 T&E Considerations for Test & Evaluation for Use Case 1

The first requirement for T&E of images is to have a sufficiently representative image collection with both individual variations, such as images during a rainstorm or images with skew, and as much as possible, combinations of individual variations, such as images with skew during a rainstorm. The T&E dataset should have operational authenticity with images collected from the same cameras, installation parameters, and environmental situations as a minimum. In addition, each category of objects to be recognized must be well-represented, including with variation in image quality. Some annotation may be needed, particularly if the output must contain bounding boxes or object position within the image. Operationally oriented testing must account for having sufficient testing data to cover:

- Statistically significant coverage in all classes of objects

- Decision boundaries with edge cases or cases outside of the bounds of the required specification

- Out-of-class object recognition

- Precision/recall preference

Metrics should be tuned to precision and recall needs of the users, as well as being calculated for low membership classes specifically. For example, some mission-critical tasks may demand very

high recall but lower rates of precision on the basis that analysts and experts must review and validate each AIES prediction. Tuning the metrics can be a challenging task since eliciting this information is difficult. Users are not accustomed to thinking in terms of error rates in their own perceptual abilities, and decision makers can lack full understanding of the performance improvement that human-in-the-loop can achieve.

For the armored vehicle recognition use case, T&E should test for operational conditions, including adversary actions to circumvent recognition.² This means testing on specified camera types, varied DPI, similar vehicles that are not armored vehicles, and weather conditions. While it is possible to use synthetic images or synthetically degraded images, operational data ensures better deployment.³

7.1.2.5 T&E Considerations in Deployment/Sustainment for Use Case 1

Deployment requires continuous evaluation and output monitoring. This is due to potential sensor failures, in this case, cameras, unanticipated or changing environmental conditions, such as a sandstorm, and to ensure the precision and recall are sufficient for operator needs. In addition, operator feedback should be tracked—at a minimum whether they accept or reject the results. Images that are rejected as wrong must have the accompanying images so that retraining can take place. Retraining needs to be reevaluated using both earlier evaluation data and the new evaluation data. Finally, if the models are run at the edge, care must be taken to build models such that they work on the deployed hardware. For the tank recognition use case, it is more likely that it would be deployed closer to the edge, so this must be accommodated, particularly if there is any image preprocessing involved.

Sustainment should continue the collection of data, which entails both an increased need for storage and planning for retraining and retesting periodically. The reasons for the requirement to retest include adversarial actions to avoid detection and domain drift where environmental changes can cause models to be more inaccurate. Not only should the deployed model be able to self-report on errors or out-of-bounds images, but also planning for annotation of new images if necessary and T&E. At a minimum, the system should be tested for “fallback” or error reporting during the lag between detection of a new image type and retraining, retesting, and redeployment.

² For instance, painting flowers on tanks to throw off recognition. Toolkits exist to support detection and mitigation .

³ For operational test, this will probably require using the armored vehicle in multiple operational environments and collecting the data real-time. Synthetic data will require validation to be used for operational test.

7.1.2.6 Summary Table for Use Case 1 - Perception

The table below provides a summary of considerations for Use Case 1 categorized by each of the five phases of the SELC.

Table 77-2. Use Case 1 – Perception Summary Table

Perception (Image Recognition)		
Requirements	Agile Methodology	- Iterative cycles, with short sprints focusing on specific features or improvements. - Regularly assessing and incorporate feedback from end-users, ensuring that the solution meets their evolving needs and remains aligned with the overall mission
	Mission Driven Use Cases	- Identifying tanks from stationary camera images, considering optimal camera angles and placement distances. - Detecting tanks in stationary camera footage, accounting for varying environmental conditions and sensor capabilities. - Analyzing historical images to address class imbalances through per-class recognition rate to ensure model is not biased and performs poorly on the underrepresented class.
	Prototyping	Augment existing pre-trained models or developing a simple proof-of-concept model to demonstrate the potential of the approach, accounting for camera requirements.
	HW & Infrastructure Requirements	- High-performance GPUs for training the model on large datasets. - Sufficient storage capacity to store training data and intermediate model checkpoints. - Scalable cloud-based infrastructure for deploying the model(s) and handling real-time inference requests, considering the integration with sensors, microphones, and cameras
Acquisition Strategy	Data Rights	Government rights to <u>datasets</u>, features, algorithms, configurations, design documentation, and trained models can both contribute to reproducibility and avoid vendor lock.
	Model Rights	The rights to models and annotations are vital to specify in order for the government to acquire and <u>retain</u> rights
	Protections for AI/Model Data	- The government should ensure that the contract specifies rights to and ability to test data pre- and post-processing tools, such as image cleaning tools. - Finally, the contract should specify that the developer provide “hooks” into the ML-based components to facilitate continuous evaluation.
	Sufficient Resources for the Full Lifecycle	- Users needed to inform on the precision/recall <u>trade</u> they seek - Resource to enable data collection and <u>curate</u> sufficient operational context and human annotation/tools. Some ML approaches require <u>pixel-level annotation</u>
Development	Data	SME Curated Data Sets used to train model
	Iterative Model Training	Validate the AI model(s) as a sub component of the larger system.
	Cards	Track results with Model, Data and Evaluation Cards to serve as authoritative guidance to DT&E
	Metrics	Algorithm performance in context of operational use cases.
T&E	List of Classes of Armored Vehicles	- Operationally oriented testing must account for sufficient testing data to cover: - Statistically significant coverage in all classes of objects - Edge cases or cases outside of the bounds of the required specification
	“Other” object categorization	Out-of-class object recognition (e.g., similar vehicles that are not armored vehicles)
	Precision/recall tradeoff	- Precision/recall preference on developed model(s) - Metrics should be tuned to precision and recall needs for users
	Camera specifications	T&E should test for operational conditions (e.g., specified camera types, varied DPI).
	Environmental requirements	T&E should test for operational conditions (e.g., similar vehicles that are not armored vehicles) and environment issues such as certain weather conditions, time of day.
Sustainment	Data Management	- Geo, orientation, time stamp - Semantic metadata/tagging - Image indexing
	Domain Monitoring	- Novel targets - Object bounding - Environmental noise - Blind zones
	Log/Artifact Management	- Visual confirmation/field engagement - Multi-sensor activation - Telemetry latency
	Algorithmic Sustainment	- Decentralized tracking/fusion - Resilience against deception/spoofing - Denosing - Super-resolution/feature augmentation - Transfer learning
	Hardware Optimization	- High-resolution sensors - Combining embedded/edge services
	User Training	Field operators and intel analysts
	Feedback Elicitation	Field engagement reports

7.2 Sensemaking (Use Case 2)

Sensemaking tasks include a wide range of data transformation, classification, and combination activities. Information extraction, machine translation (MT), social graph analysis, and event prediction are all types of applications in this category. Typically, these applications rely on ground truth (or near ground truth) reference values with which to evaluate model performance.

7.2.1 Use Case Description for Document Triage using Named Entity Recognition (NER)

Document triage specialists are charged with the task of processing electronic copies of documents that have been scanned in from hard copy that was collected during search or pulled from electronic media or feeds. In information analysis, the user does not speak the languages of the regions and must decide which documents have intelligence value. They base intelligence value on names of people, organizations, and locations.

Description: The system must convert the scanned hard copy images into electronic text and use in-language NER to identify names of people, organizations, and locations. These must then be translated or transliterated into English to allow triage analysts to make an intelligence value determination.

The use case specifications are summarized in Table 77-3.

Table 77-3. Document Triage Use Case Specifications

Specification Type	Dimension	Specification
Task-based Specifications	Type-level accuracy	Names of people, places, organizations, correctly attributed to the type.
	Type of data	Unstructured and semi-structured text converted from document images.
	Languages involved	Arabic (Modern Standard Arabic, Iraqi Arabic); English
	Reference scope	Output the full span of each reference (e.g., “Barzah Scientific Research Facility” vs. “Barzah”)
	Source provenance	System must index location of detected entities in each source document.
Workflow Measures	Current time on task	If linguist is available, a routine data collection can take two weeks to triage.
	Task frequency	Daily
	Current personnel required	To perform the task in a timely manner, would take five linguists and five triage specialists.

Specification Type	Dimension	Specification
	Recognition accuracy	0.9 f-measure
	Precision/recall tradeoff	Analysts shall be surveyed for precision/recall trade and system tuned for it.
Operating Environment	Software interaction for inputs/outputs	Multi-system application involving optical character recognition, named entity recognition, and machine translation or translation memory.
	User interaction/interface	Yes, it should provide a list of names.
	Computational resources required for task	Can be performed on a standard laptop.

7.2.2 Related T&E Impacts for Each Phase of the SELC for Use Case 2

Sensemaking systems such as described in this use case can more complex than the more focused perception use cases. This use case relies on a multi-component AIES and therefore is more complicated. The analysis of considerations yielded several themes, such as:

- As with perception tasks, boundary conditions, potential class imbalances, and specific user and operational needs must be carefully examined, in this case, related to languages, images, and domain needs.
- Requirements, data, metrics, and thresholds will need to be identified for components within the AIES.
- Data curation needs will expand since each component will have its own annotation protocol for training. T&E will require both component-level, integration-level, and deployment annotation.
- Data tracking needs for sustainment increase to add component-level results which may point to either workflow errors or component failures. These are in addition to system-level T&E needs.
- Sensemaking task T&E relies more on human involvement, both for preferences in success criteria and user interaction testing.

7.2.2.1 T&E Considerations in Requirements Definition for Use Case 2

Testers and evaluators must define sensemaking system requirements that adequately specify boundary conditions, class imbalances, and minimal operational success thresholds for the system. For sensemaking tasks, the acceptable thresholds for errors can be and often are higher than other types of tasks due to post-output user interaction and expert review. Many sensemaking tasks rely on a human-in-the-loop approach whereby users can review the model's outputs and perform manual correction and validation to provide additional feedback to the model. Additionally, sensemaking applications often operate on potentially ambiguous contexts, resulting from missing information and language-specific underdetermination. Such conditions

place a greater burden on defining requirements for explainability and confidence measures. The requirements must therefore detail the use case in terms of acceptable margins of error, the desired method and interface for facilitating user interaction, and the explainability of the model-generated results.⁴ The requirements must also formulate a framework for evaluating the trustworthiness of the whole sensemaking ecosystem, including assessments of potential data collection bias, measurements of inter-annotator agreement, and specifying the granularity and range of entity types.

7.2.2.2 T&E Considerations in Contracts for Use Case 2

Contracts must specify a sufficient number of annotators, resources for annotation, and the current availability of annotated source materials within relevant clauses of the contracts. Real-world data may be required in the form of GFI for annotation, in which case, the contract must include additional terms for acquiring, licensing, or otherwise authorizing the release of proprietary or controlled data. Additionally, since this use case supports foreign language analysis, annotators must possess verified source language fluency and/or domain expertise. A contract should accurately reflect the time and cost required for data conditioning and data annotation to support multiple reiterations of the T&E effort.

7.2.2.3 T&E Considerations in Development & Integration for Use Case 2

Predictive analytics applications tend to be user facing, although they may include AI-based components that condition the input data prior to model application, in which case, both the data conditioning and the system output should be tested. Other applications are often integrated as a single module within a larger system that feeds their output to other components of the pipeline, which may be AI-based. For instance, information extraction may provide structured data for a graph database and a different AI component would subsequently produce a graph analysis for users to review. Such pipelines require carefully specified interfaces and test harnesses that address technical and workflow-oriented issues across individual models and the overall integration of the AI-supported ecosystem. Careful attention must be given to the possibility of cascading errors, compounding uncertainty, and increasing mismatches between precision/recall preferences when AI components are integrated into a workflow environment.

Developers and evaluators should therefore produce detailed schemata of the overall sensemaking ecosystem to identify annotation standards. For this use case, the model development phase should produce sufficient high-quality annotated training data. The annotation process must follow concrete and agreed-upon annotation guidelines, which should include representative examples for all entity types as well as examples of unacceptable annotation practices to illustrate and pre-empt potential errors. Once the annotation process is complete, evaluators should compute inter-annotator agreement scores for the whole dataset as well as type-specific scores (i.e., persons, locations, and organizations) to determine whether user-generated training and evaluation reference data meet the designated reliability standards. For instance, one annotator may consider a person's title as part of their name (e.g., "Dr. Mary Jones") while another annotator may not. Well-defined annotation guidelines can prevent such inconsistencies and facilitate reproducible T&E procedures. Annotation tasks can often benefit

⁴ Explainability requirements may result from legal, ethical, or policy-related requirements, such as oversight. Policies are often written for humans rather than AI or human-assisted AI, and therefore, the requirements and resulting testing may be affected.

by adopting a “tag-a-little, learn-a-little” (TALLAL) loop in which a model is iteratively trained and tuned in multiple steps; in a TALLAL development framework, annotators produce additional training data in batches that are then incrementally fed to the model and users subsequently validate and correct system predictions, which are then once again applied to refine the model.

If integrated named entity recognition capabilities are dependent on inputs from other AI applications such as an OCR component, development, integration, and T&E requirements must address data-quality specifications at each interim throughput path between individual components. Although end-to-end evaluation should start with raw inputs and output rates, the evaluation of individual components will greatly facilitate better diagnostics and risk mitigation.

7.2.2.4 T&E Considerations in Test & Evaluation for Use Case 2

Evaluators should also collaborate with analysts to determine the minimum precision and recall metrics required of the system for their specific corpus, workflows, and operational priorities . Many NER and other sensemaking use cases often favor prioritizing higher recall over higher precision, given that it is typically more important to avoid missing true positives in named entity detection than it is to assign the correct entity type, especially when triage specialists and analysts are charged with determining a document’s intelligence value. Evaluators should also be aware that, depending on the task objective and operational priority, some entity types may require higher or lower precision rates than other types; for example, identifying named references to individual persons and places may be of greater actionable intelligence value than identifying references to organizations. More broadly, sensemaking tasks generally present varying minimum performance thresholds for different classes of categorical predictions and evaluators should document these discrepancies, both in terms of what is required and what is delivered by an AI model.

For NER tasks, evaluators should also consult with analysts and other users to determine a scoring strategy for partial matches in instances where the extent of the detected entity is different from the reference target (e.g. “al-Duhur Military Airbase” vs. “Abu al-Duhur Military Airbase”). The Message Understanding Conference (MUC-6) , CoNLL-2003 , and ACE-08 shared task guidelines and introduced several different standards for scoring complete and partial matches. For example, under the Conference on Natural Language Learning guidelines, a system must match the full extent of human-annotated named entities for both the detection and classification subtasks. Under the MUC-6 specifications, the entity detection and entity (type) classification subtasks are scored according to separate criteria: systems must match the full extent of a given named entity for entity detection subtasks, but correct entity type predictions are accepted irrespective of the extent of overlap between the reference and hypothesis pair. Lastly, the ACE scoring policy allows for partial matches if the syntactic head of the detected entity overlaps with the reference value by a minimum number of characters. The T&E team should therefore consider various approaches to scoring before committing to a particular method as part of system evaluation.

7.2.2.5 T&E Considerations in Deployment/Sustainment for Use Case 2

Given that the end goal of sensemaking applications is to facilitate analysis by human experts, evaluators and developers should consult with end users to determine how front-end interfaces and visualizations can best support their routine workflows. Users often rely on filtered, tabular

views of structured or categorized content produced by AI models, but they should also have the ability to inspect model predictions within a broader task-specific context, be it in the form of a semi-structured document with system-generated entity labels, parallel views of source-language and target-language texts in machine translation applications, or metadata-enriched transcripts of telephone communications within a graph analysis task. While a standardized question set contributes to evaluation, users should also be given the opportunity to provide their own individual comments without being bound to a set of pre-identified criteria.

Sustainable sensemaking ecosystems require predefined policies and processes for monitoring system performance and changes in the analytical domain. Retraining models on an as-needed basis, and evaluating the impact is how these should be tested. Retraining requires the collection and curation of new datasets for T&E. It is important for system requirements and development to include the continuous collection of both input data and output results for future retraining, after action review, diagnosis, and T&E.

Developers and system integrators should also enable end users to capture their corrections to model-generated predictions in a minimally disruptive manner so as not to detract from their routine roles and responsibilities. Developers should then aggregate these edits and corrections to further refine AI models and improve overall system performance. Although many industry-developed AIES include native capabilities for tuning their models with human-in-the-loop feedback, evaluators and end users should collaborate to determine whether a standalone user-oriented validation tool should be developed “in house” to capture domain-specific and language-specific targets. The provision for this should be addressed as part of the requirements definition. T&E personnel should also periodically assess how much time users spend on average when completing a discrete subtask with AIES support as an important measurement of system suitability; if, for example, users require a significantly greater amount of time to perform the same type of tasks as before, such a finding would be an additional indicator that the model may require further tuning or re-training.

7.2.2.6 Summary Table for Use Case 2 - Sensemaking

The table below provides a summary of considerations for Use Case 2 categorized by each of the five phases of the SELC.

Table 77-4. Use Case 2 – Sensemaking Summary Table

Sensemaking (Named Entity Recognition)		
Requirements	Agile Methodology	• Same as perception use case • Greater likelihood of evolving needs aligned with the overall mission.
	Mission Driven Use Cases	• Identifying and categorizing potential threats from scanned Arabic documents, such as plans for terrorist attacks, recruitment materials, or communication related to illegal activities. • Another use case could involve identifying documents containing sensitive information that could be exploited by hostile actors.
	Prototyping	• Increased need for Human Machine Teaming • Initial focus extracting Arabic text from scanned documents, followed by basic keyword-based threat identification. • Subsequent prototypes could integrate more advanced natural language processing techniques to improve the accuracy of threat detection and minimize false positives and negatives
	HW & Infrastructure Requirements	• High-performance servers capable of training AI model(s) for text recognition and threat detection, • Storage systems for managing the scanned documents and related data, intermediate results, and secure networking capabilities to ensure data integrity and protect sensitive information. • Effective scanning hardware for efficiently digitizing the hardcopy documents will be necessary.
Acquisition Strategy	Data Rights	• Should data conditioning be required for components, intellectual property (IP), data, model(s), and supporting software ownership must be specified
	Sufficient Resources for the Full Lifecycle	• Must specify a sufficient number of annotators, resources for annotation, and the current availability of annotated source materials • Since this use case supports foreign language analysis, annotators must possess verified source language fluency (reading only) and/or domain expertise • Should accurately reflect the time and cost required for data conditioning and data annotation to support multiple reiterations of the T&E effort
	RFP	• Real-world data may be required in the form of GFI for annotation, in which case, the contract must include additional terms for acquiring, licensing, or otherwise authorizing the release of proprietary or controlled data as well as ownership of annotations.
Development	Data	• SME Curated Data Sets used to train model(s)
	Iterative Model Training	• Validate the AI model(s) as a sub component of the larger system.
	Cards	• Track results with Model, Data and Evaluation Cards to serve as authoritative guidance to DT&E
	Metrics	• Algorithm performance in context of operational use cases.
T&E	System Test	• Determine the minimum precision and recall metrics required of the system for their specific corpus, workflows, and operational priorities • Determine a scoring strategy for partial matches in instances where the extent of the detected entity is different from the reference target using any specifications or guidelines (e.g., (MUC-6) , CoNLL-2003 , and ACE-08).
	Integration / System Test	• The T&E team should consider various approaches to scoring before committing to a particular method as part of system evaluation.
Sustainment	Data Management	• Corpus population & retrieval • Co-reference/NE variants • Source images (OCR use cases) • Annotations for component(s) • Source provenance metadata
	Domain Monitoring	• Unknown/novel NEs • Type ambiguity • Language/script support
	Log/Artifact Management	• Triage (analyst/linguist-in-the-loop) • Confidence scores • Component-level result(s)
	Algorithmic Sustainment	• Human (re-)annotation • Automatic language identification • Multi-lingual integration
	Hardware Optimization	• OCR devices • High-speed data xfer/HD readers
	User Training	• Guidelines for annotating training data
	Feedback Elicitation	• Support from linguists, SMEs, and analysts

7.3 Course of Action (COA) Generation (Use Case 3)

Course-of-action generation is in the class of applications where the AI-based system will consider a wide range of options in combination to achieve a goal. The options typically have constraints that help determine their viability as selections. A classic example is logistics planning where systems help determine the optimal route selections and load balancing to enable package delivery services to meet deadlines and work efficiently. Recently, COA generation has expanded to include identifying patterns-of-life from multiple non-traditional sources or to support modeling battlefield possibilities.

The use case selected in this category is COA recommendation for military mission planning training. Note that this use case must be generally specified in an unclassified document. A more detailed use case will be developed in future work.

7.3.1 Use Case Description for COA Recommendation for Mission Planning

Commanders performing mission planning must determine planned courses of action for upcoming troop movements and actions. Any mission plan must be investigated for potential effects and adversarial responses, and multiple plans may be contrasted to determine the space of available outcomes. The models in the system generate COAs for the commander to assess. In MDO where speed and complexity are both increased, the training for commanders to make accurate assessments and mission plans is essential for them to be able to respond on the battlefield.

Description: The system must take an existing assumed state of the battlefield as generated by modeling and simulation, and either generate potential courses of action or analyze suggested courses of action. The use case specifications are summarized in Table 7-5.

Table 7-5. COA Recommendation Use Case Specifications

Specification Type	Dimension	Specification
Task-based Specifications	Information required?	Information about the current state of the battlefield and mission objectives - typically the output of one or more M&S systems which simulate things such as radar systems or weapon capabilities and trajectories. Inputs from M&S systems which generate sample mission states and objectives.
	Type of data	Troop placements, battlefield topology, important areas (e.g., minefields, build-up-areas), troop strength, available equipment, and weather. This must be represented for both red and blue forces.
	User interaction - display	The display must show the suggested courses of action in an easy-to-understand format—for mission planning, this is often a geographic or spatial display. If multiple options are generated by the AI, it is important to support contrast of different options.
	User interaction - update	If the state of the world changes, the options must be updated.
	User interaction - explainability	The system should explain the data used to support the suggested mission options.
	System response time	System will generate first step within 2 minutes and remaining plan within 3.
Workflow Measures	How much time is currently spent on task?	TBD - manual scenario generation when conditions change
	How often is task done?	TBD - anywhere from hourly to weekly
	How many people are required to do task?	Multiple
Operating Environment	Does the software interact with other systems?	Yes, it interacts with modeling systems.
	Does the software interact with the user directly?	Yes
	Describe the computational resources required for operation	Standard PC configuration

7.3.2 Related T&E Impacts for Each Phase of the SELC for Use Case 3

AIES that use generative approaches increase complexity of testing over sensemaking system. This use case is a course-of-action generation task. Note that active learning systems that adapt

over time were not considered here. The analysis of this use case shows the increased, and often more involved, considerations such as:

- Input data for development and testing often involves generated data from modeling systems as well as other AIES. This means closer data analysis during the T&E process.
- COA generation systems are very dependent on user interaction. In this use case, the system goals or parameters may change during operations which can increase the conditions for testing significantly.
- User interface and human judgment factors are critical for success.
- Since COA generation systems rely on SME to define criteria and inform bounds, their involvement throughout the lifecycle is critical for both T&E and tracking intermediate results is both more complex and more necessary.

7.3.2.1 T&E Considerations in Requirements Definition for Use Case 3

A difficulty in specifying requirements for the COA generation use case is the dependency on a potentially rapidly changing environment and HMT aspects. At a minimum, requirements should specify the user base, the method of communication with the users, and user interface. A common practice is to work with military SMEs to develop relevant objective functions for ranking alternative COAs, which aid in capability development. This can be challenging where human judgment factors come into play, such as commander's intent. They should also specify methods for both updating the parameters of the problem quickly and efficiently, the response time to these changes, and methods for saving and tracking both results and user feedback. Because of human involvement, requirements will focus more on system behavior, explainability, and interface. It will also require specifying testing using human judgments as part of the evaluation.

7.3.2.2 T&E Considerations in Contracts for Use Case 3

Because evaluation is highly dependent on user engagement, contracts must be set up to ensure a sufficient number of testers and/or subject matter experts to carry out evaluations so that results are reliable and statistically significant. In addition, since human-involved testing represents a scheduling challenge, this should be planned for. Contracts should also specify the range of mission scenarios in T&E and the M&S support used to both train and test the AIES. If the M&S is not government owned, purchase can represent a significant budget item.

7.3.2.3 T&E Considerations in Development & Integration for Use Case 3

COA generation, unless integrated into autonomous systems, is generally user-facing. The integration challenge often arises when systems provide unexpected results. For instance, if deep reinforcement learning (DRL) is used, the most efficient solution to a problem may be an untenable one, such as "doing nothing" or "stopping the game." Since these untenable problem solutions can and have arisen, domain SMEs should participate in iterative development and refinement. Additionally, if the COA generation relies on external data sources, such as from sensors, the interfaces and ability to update are critical. T&E, therefore, must consider individual components, their interfaces and ability to detect or mitigate faulty information exchanges, and the potential for cascading errors. The T&E role in diagnosis and mitigation becomes especially important for a successful system. Finally, the ability of the system to be able to convey

information such as explanation or planning steps in a meaningful way to the user must be considered in integration testing.

7.3.2.4 T&E Considerations in Test and Evaluation for Use Case 3

There are many different T&E considerations for mission planning. Mission planning represents a greater challenge for designing T&E protocols and automation due to the lack of a single right answer, and in fact, sometimes situations where acceptable answers change dramatically in response to small input changes. In isolation, we can consider the quality of a presented solution. We can separate out three T&E aspects:

1. **Checks for obviously poor outcomes.** We can look for solutions that are wrong in a way that is noticeable by a non-expert, or which violates known rules of thumb. For example, in mission planning, it is generally given that minefields should never be placed near built-up areas (e.g., cities), and a COA generator that produced such a result would be immediately suspect.
2. **Rate outcomes by experts.** Solutions can be judged by SMEs in the field to determine if they are reasonable outcomes, and if an ordering is provided by the system, the SMEs can also rate the relative ordering for reasonableness. T&E could also include assessments of whether the system altered their thinking because it presented a novel solution with a sound explanation.
3. **Evaluate for range of options.** When a COA generator creates multiple outcomes, one of the hardest things to evaluate is not just the quality of each result individually, but the quality of the option set. For example, does it provide a good range of options, and not just a few minor variants of one another?
4. **Checks for robustness.** Popular AI techniques such as DRL can produce results that perform extremely well under the brittle assumption that the M&S they interact with is a perfect representation of reality. A minor inaccuracy can result in dramatic performance degradation. COA generation techniques must be robust against the “reality gap” and T&E must reflect this.

User interfaces and explanation testing is also heavily reliant on user assessment. A standard evaluation technique is to design rating scales with sufficient evaluators to ensure a significant or accurate finding. Inter-evaluator agreement testing may also be employed as a strategy.

7.3.2.5 T&E Considerations in Deployment/Sustainment for Use Case 3

During deployment, user evaluation of the deployed solution will be important. Testing of the user interface and explanation testing can be conducted along the same lines as during the T&E phase, but in the context of the operational environment. In addition, trust metrics may also be needed to get a broader understanding of whether and under what conditions users will rely on the system, sometimes called calibrated trust.

Sustainment for COA generation relies on updating and vetting knowledge for changing battlefield situations and, potentially, goal states. Additionally, M&S required for COA generation supports testing. Because of this, a greater human burden exists for T&E. For instance, if a goal state changes, previous goal states or subgoals may be entirely irrelevant. At the same time, there may be a desire to maintain those as possibilities and the knowledge

represented for generation to persist. Sustaining the M&S and verifying its accuracy when adversaries can change the rules of the game with new technologies, tactics, partners, priorities, etc. represents a challenge since T&E for M&S can be too slow to respond to changing battlefield conditions. This means the AI-based COA evaluations can be inaccurate since they are based on outdated M&S. Simple regression testing is not sufficient, and a more considered approach could be necessary. Finally, testing for the ability of the user to update the system may be a critical consideration or testing challenge.

7.3.2.6 Summary Table for Use Case 3 – Course of Action

The table below provides a summary of considerations for Use Case 3 categorized by each of the five phases of the SELC.

Table 77-6. Use Case 3 – Course of Action Summary Table

Course of Action (COA) Generation		
Requirements	Agile Methodology	Requires continuous refinement of the ML model(s) based on user feedback and iterative development. Working with SMEs to develop relevant objective functions for ranking alternative COAs can aid in capability development, but this can be challenging when human judgment factors are involved.
	Mission Driven Use Cases	Creating realistic use cases that consider various factors such as urban or rural environments, asymmetric warfare, or counterinsurgency operations can help ensure that the ML algorithm generates COAs that are relevant and effective for a wide range of missions. Increased need to focus more on system behavior, explainability, and interface, and will require testing using human judgments as part of the evaluation.
	Prototyping	Extensive use of Human Machine Teaming because human expertise is vital to make informed decisions that consider factors such as strategic goals, risks, and collateral damage Rapid prototyping involves creating early versions of the ML model(s) to test effectiveness and gather feedback from end-users Need to quickly identify areas for improvement and iterate on the design based on user feedback, system behavior, and explainability
	HW & Infrastructure Requirements	Requires powerful GPUs for training the model, edge computing devices for on-site processing, and secure communication networks to ensure the safe transmission of sensitive data between the algorithm and human decision-makers. Involve complex simulations, large-scale data processing, and computationally intensive algorithms for generating and evaluating potential courses of action.
Acquisition Strategy	Model Rights	Government rights to <u>datasets</u>, features, software, model(s), configurations, design documentation, and trained models can both contribute to reproducibility and avoid vendor lock.
	Sufficient Resources for the Full Lifecycle	- Users needed to inform on the levels of explanation they seek to inform evaluation plan - Resource to enable data Collection and <u>curate</u> sufficient operational context and human annotation/tools. Some ML approaches require <u>pixel-level annotation</u>
Development	Data	SME Curated Data Sets used to train model
	Iterative Model Training	Validate the AI model(s) as sub-component(s) of the larger system.
	Cards	Track results with Model, Data and Evaluation Cards to serve as authoritative guidance to DT&E
	Metrics	Model performance in context of operational use cases.
T&E	User interaction - Display	Checks for obviously poor outcomes: e.g., COA generator placing minefields near built-up areas like cities, a suspect result Evaluate for range of options: not just the quality of each result individually, but the <i>quality of the option set</i>. For example, does it provide a good range of options, and not just a few minor variants of one another?
	User interaction - Update	Checks for robustness: A minor inaccuracy comparing to M&S representations can result in dramatic performance degradation. T&E needs ensure COA generation techniques must be robust against the “reality gap”.
	User interaction – Explainability	Rate outcomes by experts: Solutions can be judged by SMEs in the field to determine if they are reasonable outcomes, and if an ordering is provided by the system, the SMEs can also rate the relative ordering for reasonableness. Soundness of SME decision: T&E could also include assessments of whether the system altered SME’s thinking because it presented a novel solution with a sound explanation.
Sustainment	Data Management	- Aggregation of force levels, field environment, fires, collateral risk, JIIM capabilities, etc. - Digital shadows and digital twins
	Domain Monitoring	- Operational uncertainty - Partial observability
	Log/Artifact Management	Markov-decision processes (MDPs), states and transition probabilities
	Algorithmic Sustainment	- Realignment of goal states/outcomes with RL policies and rewards - Sub-model optimization and eval
	Hardware Optimization	- Secure encrypted field devices - Parallel computing architectures (edge/embedded/hybrid integration)
	User Training	Facilitation for advanced HMT
	Feedback Elicitation	Debriefing operators, analysts, command

7.4 Summary of Use Case Analysis

In Section 7, three types of AIES were described to highlight that further implications need to be accounted for with AIES Testing depending upon the specific type of use cases: perception, sensemaking, and course of action generation. It then expressed the implications for each phase of the SELC specific to the use cases. The perception use case presented image recognition of armored vehicles and highlighted the data-related challenges for ML development. The sensemaking use case was named entity recognition for document triage and showed how the integration of two application types into a single capability causes extra needs for T&E. The course of action generation use case related the use of COA generation in mission planning exercises. It emphasizes the human-machine teaming element of T&E.

While each use case or even category of use cases has one or two unique aspects relevant to the types and amount of data and appropriate algorithms, there are common themes to be found that carry across all use cases. The requirements must have sufficient detail with fidelity to the mission to facilitate accurate testing. Contracting must account for the collection and curation of data along with providing for the resources to accomplish both development and testing. Development and integration focus on specifying clear APIs and understanding the difference between algorithm performance and system effectiveness. Integration testing often focuses on the potential for error propagation. Mission-oriented T&E can demand both computational and human resources. Computational resources are primarily the need for annotated ground-truth or approximate ground-truth. Human resources are required when the AI-enabled component presents results directly to the user. Deployment T&E must account for the use of the AI-enabled component in situ, addressing user interaction and result utility. Sustainment T&E is more continuous and must include facilities to track results allowing updates to adapt previous data, incorporate new data, and, potentially, redefine performance metrics based on operational use.

Table 7-7 shows graphically one set of dimensions. Perception tasks tend to require huge amounts of painstakingly curated data for development and T&E, however, benefits from having a single right answer for the model training and test. Sensemaking tasks are often a combination of problems with ground truth and problems without it. Generally, this means that curation is straightforward and curating the data and results requires a lower level of effort than with perception tasks. In fact, ground truth can be built into the data itself for predictive analytic tasks. Course of action generation requires less annotated data, and depending on the problem itself, potentially none. At the same time, there is a wide range of possible right answers, and the evaluation burden is with obtaining expert judgements.

Table 7-7. Use Case Differentiation

	Perception	Sensemaking	Course of Action Generation
Data Complexity	Low	Very Low	Extremely High
Curation & Annotation Complexity	Low	Medium	Low
Level of Effort	High	Medium	Very High
% of Single Right Answer	High	Medium	Low

Appendix A Adaptive Acquisition Framework from DoD 5000.89

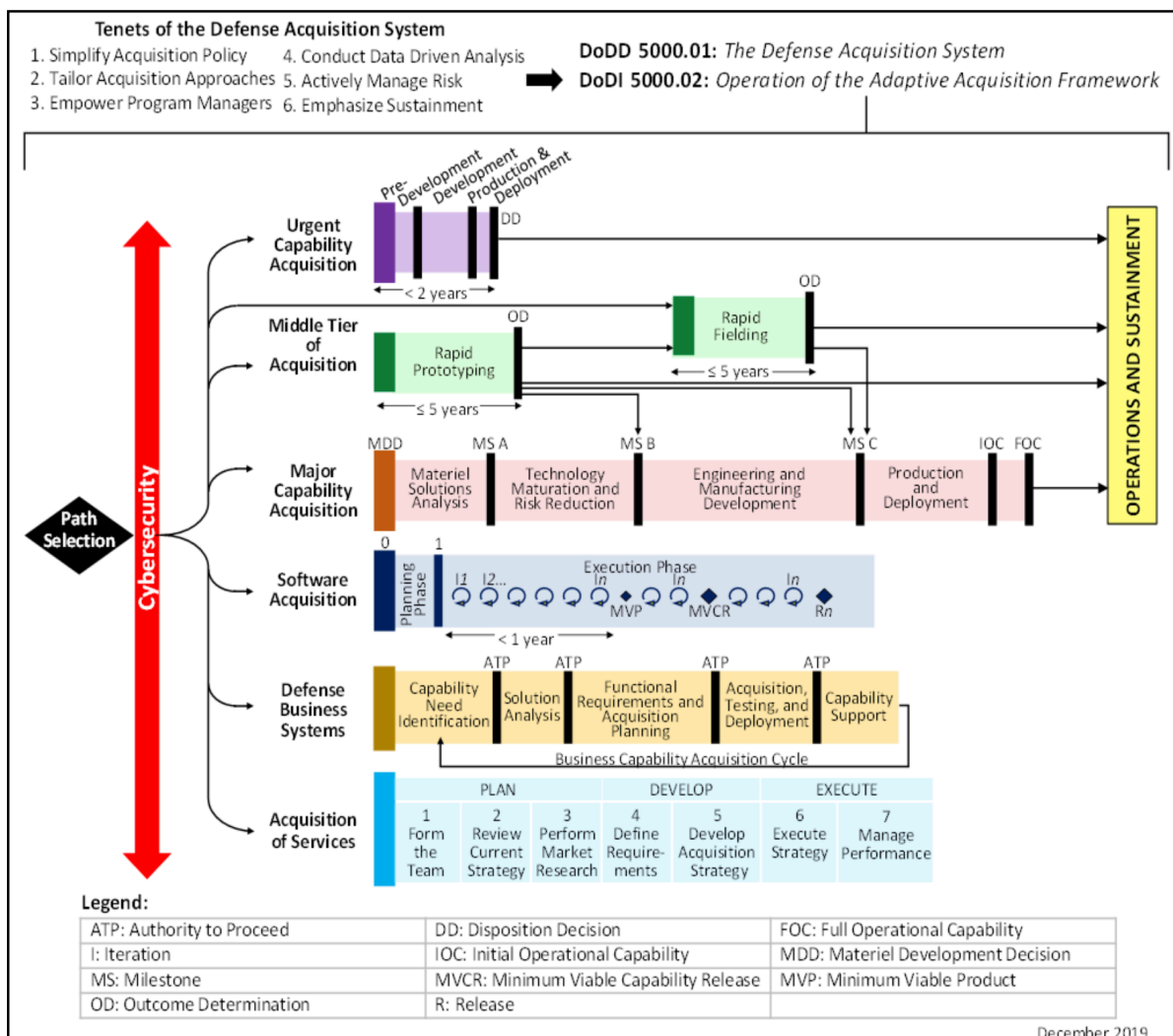


Figure A-1, Adaptive Acquisition Framework from DoD 5000.89

The software acquisition pathway has two phases: Planning and Execution.

Planning Includes

Requirements, acquisition strategy (contracting) but also includes IP strategy, test strategy, cost estimates and lifecycle sustainment planning.

Execution Includes

Development and test with an iterative approach including sustainment starting with a Minimum Viable Product (MVP) and maintenance and a continuous state of monitoring.

Programs developed using the Software Acquisition Pathway must “demonstrate the viability and effectiveness of capabilities for operational use” within one year of funding. After the first year, new operational capabilities must be delivered at least annually, but “more frequent updates and deliveries are encouraged where practical.”

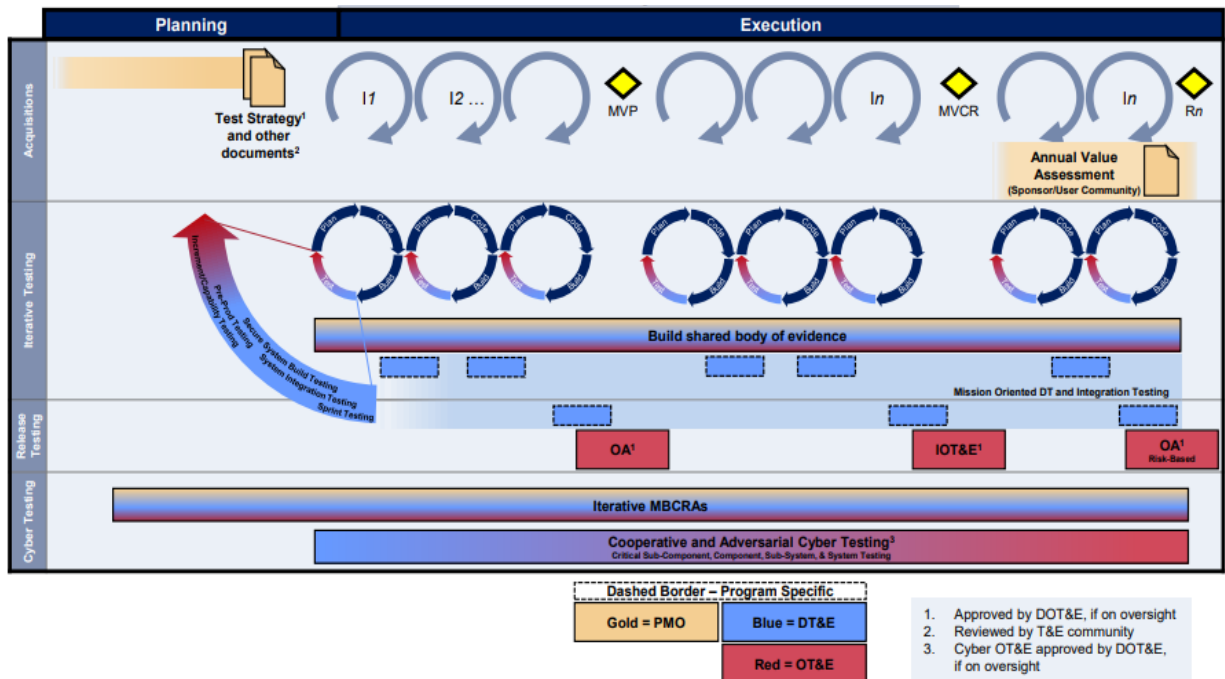


Figure A-2. T&E for the Software Acquisition Pathway [35]

Appendix B Key Differences between Traditional Software and AI Identified in GAO-22-104765

Table B-1. Key Differences between Traditional Software and Artificial Intelligence

	Traditional software	Artificial Intelligence (AI)
Developing	- Static code is the primary input for traditional software. These are instructions written by a human for machines to execute. These instructions may include algorithms—a series of specific instructions to execute under specific conditions. - Data are not necessary to develop traditional software. - Running traditional software requires variable computing power. For example, a simple software program like a word processor does not require significant computing power, but a command-and-control tool that has performance requirements to render high-quality images and simulate real-world conditions would.	- AI-specific algorithms are one of the two primary inputs for an AI model. Available AI algorithms can produce a range of desired outcomes such as classifying images or providing a prediction based on patterns in data. Rather than codify explicit instructions for a computer to process, an AI developer creates or selects one or more AI algorithms that outline how the computer will learn on its own from the data. - Data are the other primary input for an AI model. Large sets of data are required to train the AI model. For example, to train an AI model to recognize a target such as a car, an AI developer must feed it thousands of car images so that the AI model can learn all the aspects of what a car looks like to identify one on its own. - Significant computing power is necessary to train and store data for an AI model. Specifically, an AI model typically requires access to high performance computing or cloud computing to conduct complex calculations that are otherwise difficult to do on a single computer. Cloud computing harnesses the ability to temporarily access servers to support such calculations while also moving and storing vast amounts of data necessary to train an AI model quickly.^a
Using	- Traditional software is deterministic in that it behaves the same way every time until updated by a human. This means that the same inputs would result in the same outputs regardless of how many times the software program is run. For example, software that generates a location status report from a personnel database will generate the same report each time unless a human enters new data into the database or changes the code that creates the report. - Traditional software is periodically tested for errors—known as debugging—once deployed. The debugging process detects anomalies or unintended inputs in the code that may cause software to crash or not run as intended. The deterministic nature of traditional software makes it easier to identify and correct the source of errors.	- AI is dynamic and adaptive, not always deterministic. In some cases, AI algorithms have an element of randomness such that the same or very similar inputs may generate different outputs. AI algorithms produce probabilistic outcomes, meaning the output of the AI model is a prediction, not a certainty. Moreover, retraining or adapting an AI model based on new data leads to different outputs for given inputs—the desired result of such adaptation being improved performance. - Entities—such as the AI developer and end users—should continuously monitor the AI model to ensure it is working as intended after it is deployed. If the data used to train an AI model do not match the data being used to run the model, it can result in model drift and performance degradation. For example, if a predictive maintenance model is trained on data from vehicles in desert conditions and then applied to vehicles in non-desert conditions, model performance may suffer. The non-deterministic nature of AI can make it difficult to identify a point of failure or pinpoint why an AI model produced the wrong result.

Source: GAO representation of Department of Defense and other subject matter expert provided information. | GAO-22-104765

Note: AI refers to machine learning capabilities—the predominant type of AI being developed today.

^aAs defined by the National Institute of Standards and Technology, cloud computing is a means for enabling on-demand access to shared pools of configurable computing resources.

Appendix C Ten Themes for a T&E Methodology Software and AI

Dr. Laura Freeman, the Director of the Intelligent Systems Lab at Virginia Tech, has developed ten themes for a test and evaluation methodology to assure systems will function as an autonomous system . Descriptions and Applications have been added to these themes to support this maturity and set of T&E practices.

Table C-1. Ten Themes for Test and Evaluation Methodology Software and AI⁵

Ten Themes	Description	Application(s)
<u>Theme 1:</u> Testing and Evaluation is a Continuum	AI testing is an iterative and recursive process requiring support over the lifecycle.	As AI develops products, testing are characterized by iterative loops over the lifecycle where resources are allocated over the lifecycle.
<u>Theme 2:</u> The Continuum Does Not End Until the System Retires	Staff resources and support actions are sustained over the product lifecycle to disposal or retirement.	This is a change in mindset requiring AI support by technical staff and services.
<u>Theme 3:</u> Integrating information from disparate data sources requires methods	Recognition of disparate paths for data is part of the evaluation of methods and applications.	Exploratory searches to find data from a range of sources lead to new applications, innovations, and integration.
<u>Theme 4:</u> Data management is foundational	Data methods, tools, and disciplines are applied to ensure quality in data management.	Mature data management and governance with best practices.
<u>Theme 5:</u> AIES require a risk-based test approach	Different test approaches integrate risk into methods and practices.	Risk management is understood across staff and part of test methods and practices.
<u>Theme 6a:</u> Previous test metrics still apply, but they may have different interpretations	Test metrics are understood and comprehensive to measure efficiency and effectiveness performance over the lifecycle.	Develop and compare with best practices for test metrics.
<u>Theme 6b:</u> We need new metrics focusing on risks for AIES	Prevailing performance measures reflect potential risk attributes over the lifecycle.	Development and use of risk metrics correspond to the T&E lifecycle.
<u>Theme 7:</u> An expanded definition of threat is necessary (adversarial AI)	Qualify and expand threat landscape of potential security risks.	Search and expand portfolio of terms on AI and T&E threat, security, and privacy terms and how they are tested.
<u>Theme 8:</u> Operational relevance is essential	Assess operational practices baseline for T&E activities leading to mature operations.	Characterize, formalized, and conduct operational performance evaluations with testing in

⁵ Reformatted from

Ten Themes	Description	Application(s)
		developed AI products. Conduct benchmarks with academia and commercial organizations.
<i><u>Theme 9:</u></i> All AI areas need Testbeds	Evaluate current test environments and compare other lab testbeds for effectiveness.	Survey other test labs for AI methods, tools, and techniques from their test labs. Survey staff on current limitations.
<i><u>Theme 10:</u></i> T&E workforce	Evaluate T&E baseline of skills and competencies and compare what is needed to mature staff capabilities.	Develop training paths for developers and quality test engineers. Evaluate staff core competencies and deficiencies. Build collaborative team cultures coupled with tools and techniques as best practices.

Appendix D Evaluation Card Template

The Evaluation Card specifies why the evaluation is being done: the use cases, the purpose, and the information needed. After this, it delineates the specifics of the evaluation itself, starting with the evaluation conditions, such as the protocols for data handling and exchange, which will be different for internal and external evaluations; metrics; annotation needs and standards; and scoring thresholds. In addition, links to Model Cards, system descriptions, and datasheets provide the ability to identify and recreate evaluations. The evaluation results can also be captured through datasheets and lessons learned, enabling future evaluations to benefit from any challenges encountered during an evaluation.

An Evaluation Card should contain, at a minimum, the following elements:

1. Evaluation Purpose: system selection, system improvement measurement, system readiness, system trustworthiness
2. Task Overview: use case specification link
3. Evaluation Goal: information need being measured
4. Evaluation Conditions
 - a. Protocol for handling data (training/evaluation split)
 - b. Protocol for data exchange (if outside parties)
 - c. System specification under which the evaluation occurs (hardware, software, communication protocol)
 - d. Evaluation dates
5. Data curation and labeling specifications
 - a. Ground-truth or not
 - b. Specification of annotation standards for ground truth
 - c. Specification of data acquisition/curation for non-ground-truth (MT, summarization)
6. Scoring Metrics
 - a. Automatic calculation – metrics that are calculated by checking against ground-truth and can be scored with tools
 - b. Human judgement criteria – metrics that require human judgement/assessment of the quality or usefulness of the system results
7. Scoring Thresholds
 - a. If for “go/no-go” decision, minimum threshold specifications or error tolerances
 - b. For multi-metric evaluations, specify if any carry more weight
 - c. Links to KPP/KSA
8. Link to system descriptions
9. Link(s) to Model Cards (See Appendix E)
10. Link(s) to Evaluation Card (See Section 6.1.1.1)
11. Lists of testing support scripts
12. Links to evaluation result data – raw and analyzed
13. Description of lessons learned
14. Points of contact

Table D-1. Sample Evaluation Card for Use Case 2

Field	Subfields	Value
Evaluation Purpose		Integration readiness of triage capability for workflow selection team.
Task Overview		The system must convert the scanned hard copy images into electronic text and use in-language NER to identify names of people, organizations, and locations. These must then be translated or transliterated into English to allow triage analysts to make an intelligence value determination.
Evaluation Goal		Assess the tool for accuracy in its ability to support triage analysts.
Evaluation Conditions	Protocol for handling data	DoD will perform evaluation internally and acquire data from operational unit.
	Protocol for data exchange	N/A
	System Specification	• Software should run on standard configuration of Windows PC. • Software should accept images from standard off-the-shelf scanners.
	Evaluation Dates	TBD
Data curation/labeling specifications	Ground-truth or not?	Optical character recognition (OCR) – yes Named entity recognition (NER) – yes Machine translation (MT) of non-named entities – no Triage decision – yes
	Annotation standards	Optical character recognition needs transcription of the text in the image. Images will be zoned and only text counts. Named entity recognition requires annotation guidelines (Link here) Machine translation requires reference translations Priority binning for triage decision – 3 priorities NOTE: for the first 3, Arabic linguists will be required
	Data acquisition / curation	Images from scanners with transcription of images Named entities annotated in transcribed text Translations require translators to provide simple translations Triage binning decisions performed by analysts

Field	Subfields	Value
Scoring Metrics	Automatic calculation	OCR – Word error rate NER – precision/recall per named entity category (allow partial matches). MT – BLEU as well as focus on named entity translation/transcription Triage – binning precision/recall per category.
	Human judgment criteria	None at this time
Scoring Thresholds	Minimum threshold	Precision/recall in binning at 0.8. Binning for top priority at 0.9.
	Multi-metric	OCR – WER at 0.15 NER – f-measure of 0.9 MT – BLEU, for NE f-measure of 0.8
KPP/KSA Link		Link to requirements doc
System Description Link		File location in /usr/dev
Model Card Link(s)		Link to model card repository
Data Card Link(s)		Link to data card repository
Test Support Script(s) Link		Test support scripts will run function, calculate results on output. Each exists, such as Tesseract suite for OCR, NIST NER scoring, BLEU (from NIST), and Excel for triage decision
Results Link		Link to script outputs
Lessons Learned		TBD – specified at end of evaluation.
POCs		Pat Smith

Appendix E Model Card Template

Model Cards have undergone some greater formalization since the paper’s original specification. The most current available version (v. 0.0.2)⁶ has fields that are categorized, named, and defined as reproduced below in plain text; these appear to be a useful basis from which to grow. These are captured as properties in a JSON schema. Below is a bulletized exploration of that schema, and this is used as a foundation of the model described in the implementation recommendations.

I. Model Details

- Name (String): The name of the model
- Overview (String): A brief one line description of the model
- Documentation (String): A thorough description of the model and its usage, including a set of operational parameters, which is also referred as “Operating Envelope” in industrial operations . This provides information on levels of confidence on quality performance measures (e.g., certain Key Performance Indicators (KPIs) are optimized vs. others are still under improvement) for users to trust and accept the model.
- Owners (Array of Owner): Individuals or teams who own the model
 - Owner (Object):
 - Name (String): Name of the owner
 - Contact (String): Contact information of the owner
- Version (Object): Version of the model
 - Name (String): Name of the version
 - Date (String/Date): The date the version was released
 - Diff (String): The changes from the previous version
- License (Array of License): The model’s license for use
 - License (Object):
 - Identifier (String): A standard SPDX license identifier (<https://spdx.org/licenses/>), or \"proprietary\" for an unlicensed module
 - Custom Text (String): A text of a custom license
- References (Array of Reference): Links providing more information about the model
 - Reference (Object):
 - Reference (String/Uri): A reference to a resource
- Citations (Array of Citation): How to reference this Model card
 - Citation (Object):
 - Style (String): The citation style
 - Citation (String): The citation itself

⁶ https://github.com/tensorflow/model-card-toolkit/tree/main/model_card_toolkit/schema/v0.0.2

II. Model Parameters

- Model Architecture (String): The architecture of the model
- Data (Array of Dataset): The datasets used to train and evaluate the model
 - Dataset (Object):
 - Name (String): The name of the dataset
 - Link (String/Uri): A link to the dataset
 - Sensitive (Object):
 - Sensitive Data (Array of String): A description of any sensitive data in the dataset
 - Graphics (Object):
 - Description (String): A description of this collection of graphics
 - Collection (Array of Graphic): A collection of graphics
 - Graphic (Object):
 - Name (String): Name of the graphic
 - Image (String/base64): The graphic encoded as a base64 string
 - Description (String): A description of the dataset; can describe size of dataset, whether it is used for training, testing, or validation, etc.
- Input Format (String): The data format for inputs to the model
- Output Format (String): The data format for outputs from the model

III. Quantitative Analysis

- Performance Metrics (Array of Performance Metric): The model performance metrics being reported
 - Performance Metric (Object):
 - Type (String): The type of performance metric
 - Value (Number): The value of the performance metric
 - Slice (String): The name of the slice this metric was computed on; by default, assume this metric was not sliced
- Graphics (Array of Graphic): *see above*

IV. Considerations

- Users (Array of User): Who are the intended users of the model?
 - User (Object):
 - Description (String): A description of the user
- Use Cases (Array of Use Case): What are the intended use cases of the model?
 - Use Case (Object):
 - Description (String): A description of the use case
- Limitations (Array of Limitation): What are the known technical limitations of the model? (e.g., what kind of data should the model be expected to *not* perform well on? What factors might degrade the model performance?)
 - Limitation (Object):
 - Description (String): A description of the limitation
- Tradeoffs (Array of Tradeoff): What are the known tradeoffs in accuracy/performance of the model?
 - Tradeoff (Object):
 - Description (String): A description of the tradeoff

- Ethical Considerations (Array of Risk): What are the ethical (or environmental) risks involved in the application of this model?
 - Risk (Object):
 - Name (String): The name of the risk
 - Mitigation Strategy (String): Strategy used to address this risk

Table E-1. Model Card Example for Use Case 2⁷

Name	Fine Grained Named Entity Recognition
Registered Name	crf_tagger
Model Type	BiLSTM
Description	This model identifies a broad range of 16 semantic types in the input text. It is a reimplementaion of Lample (2016) and uses a biLSTM with a CRF layer, character embeddings and ELMo embeddings.
Primary Uses	Unknown
Last Updated	2020-06-24
Version	1
Relevant Task	ner
Predictor Name	sentence_tagger
Performance Measure	Accuracy and Span-based F1 metric
Unitary Results	On the validation set: Accuracy: 97% F1: 88%
Evaluation Dataset	Ontonotes 5.0
Training Dataset	Ontonotes 5.0
Training Motivation	None
Training Config	https://raw.githubusercontent.com/allenai/allennlp-models/main/training_config/tagging/fine-grained-ner.jsonnet
Training Preprocessing	None
Archive File	https://storage.googleapis.com/allennlp-public-models/fine-grained-ner.2021-02-11.tar.gz
Developers	Lample et al
Contributors	Unknown
Paper	Neural Architectures for Named Entity Recognition @article{Lample2016NeuralAF, title={Neural Architectures for Named Entity Recognition}, author={Guillaume Lample and Miguel Ballesteros and Sandeep Subramanian and K. Kawakami and Chris Dyer}, journal={ArXiv}, year={2016}, volume={abs/1603.01360}}
Contact	allennlp-contact@allenai.org
Caveats	None
Ethical Considerations	None
Install Instructions	<code>pip install allennlp==2.1.0 allennlp-models==2.1.0</code>

⁷ <https://demo.allennlp.org/named-entity-recognition/fine-grained-ner>

Appendix F Data Card Template

Table G-1 below shows an example of Google’s data card. The data owner/provider fills out a Data card, collecting important aspects and relevant information of the dataset. Based on the prior JAIC (transitioned to CDAO) pilot exercises and lessons learned, it is recommended to organize the Data card into at least three primary parts:

Primary Information about the Dataset:

This is the basic information about the dataset, allowing the team to understand the purpose of the dataset, including following items:

- Period of data collection, intended dataset usage, and relevant timeframe for the dataset utilization if any
- File source, size, licenses, and data fields
- Storage location and frequency of updates
- Assumptions and limitations of the dataset
- Data sanitization method for the use of PII/PHI
- Potential privacy and ethical risks, especially related to the PII/PHI data
- If the data is labeled, the methodology for labeling the data, and labeling validation, if applicable

Statistics Population of the Dataset:

This part of the Data card should describe the important aspects of the data and provide a high-level description of what the data represents, including metadata fields and all possible classes. It should also describe how the data was collected and in particular if there are known errors or gaps in the dataset (as applicable), and describe the ontology of the data—with a source link, if possible.

This type of information should be collected and organized before any data collection actually begins. It is also desirable to include a breakdown of the distribution of categories/classes in the population (if feasible at this point).

The population of data and details of the distribution of categories/classes will benefit the data splitting efforts to assign specific percentage into training, validation, and test datasets for the ML development.

Throughout the cycles of data ingestion, this population section should be consistent and does not require changes in further revisions of the Data card, unless there are significant changes that affect the overall population that the data represents.

Sample Dataset:

This section will be the sample itself, properly representing the population of the dataset. This should be updated with new revisions as samples are added to the dataset. If applicable, revised samples should also be reflected in the distribution of classes for that particular sample.

Detailed below are recommendations for fields appropriate to include in a Data card :

- I. Dataset Source Information:
 - Data Source Name
 - URL
 - Access
 - Data Type
- II. Purpose of Dataset
 - For what purpose was the dataset created?
 - Who created this dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)?
 - Over what timeframe was the data collected?
 - Will older versions of the dataset continue to be supported/hosted/maintained?
 - Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)?
- III. Distribution of Dataset
 - Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created?
 - How will the dataset be distributed?
 - When will the dataset be distributed?
 - Will the dataset be distributed under a copyright or other IP license, and/or under applicable terms of use?
 - Have any third parties imposed IP-based or other restrictions on the data associated with the instances?
 - Who is supporting/hosting/maintaining the dataset?
 - How can the owner/curator/manager of the dataset be contacted (e.g., email address)?
- IV. Composition of Data:
 - What do the instances that comprise the dataset represent (e.g., documents, photos, people, countries)?
 - How many instances are there in total?
 - Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set?
 - What data does each instance consist of?
 - Grades served? Are relationships between individual instances made explicit (e.g., users' movie ratings, social network links)?
- V. Data Confidentiality:
 - Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctor-patient confidentiality, data that includes the content of individuals non-public communications)?
 - Does the dataset identify any subpopulations (e.g., by age, gender)? If so, please describe how these subpopulations are identified and provide a description of their respective distributions within the dataset.

- Is it possible to identify individuals (i.e., one or more natural persons), either directly or indirectly (i.e., in combination with other data) from the dataset? If so, please describe how.
- Does the dataset contain data that might be considered sensitive in any way (e.g., data that reveals racial or ethnic origins, sexual orientations, religious beliefs, political opinions or union memberships, or locations; financial or health data; biometric or genetic data; forms of government identification, such as social security)?
- If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (e.g., were individuals in question told that their data would be retained for a fixed period of time and then deleted)?

VI. Data Collection:

- How was the data associated with each instance acquired?
- What mechanisms or procedures were used to collect the data (e.g., hardware apparatus or sensor, manual human curation, software program, software API)?
- If the dataset is a sample from a larger set, what was the sampling strategy (e.g., deterministic, probabilistic with specific sampling probabilities)?
- Was data collected from the individuals in question directly, or obtained via third parties or other sources (e.g., websites)?
- Were the individuals in question notified about the data collection?
- Did the individuals in question consent to the collection and use of their data?
- If consent was obtained, were the consenting individuals provided with a mechanism to revoke their consent in the future or for certain uses?

VII. Pre-process of Data:

- Was any preprocessing/cleaning/labeling of the data done?
- Was the “raw” data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)?
- Is the software used to preprocess/clean/label the instances available? Has the dataset been used for any tasks already?

VIII. Data Repository:

- Is there a repository that links to any or all papers or systems that use the dataset?
- What (other) tasks could the dataset be used for?
- Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?
- Are there tasks for which the dataset should not be used?
- Is there an erratum?
- Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)?

Table F-1. Data Card Example for Use Case 1⁸

Open Images Extended - Crowdsourced Open Images Extended - Crowdsourced intends to capture global representation. This dataset comprises over 478,000 images and associated labels from otherwise under-represented populations. It can be used with Open Images V4.		
PUBLISHER(S) Google LLC	INDUSTRY TYPE Corporate - Tech KEY APPLICATION Machine Learning, Object Recognition	INTENDED USE CASE(S) - Identify objects or context of photos visually (e.g., through Lens or Camera) - Find objects, plants, animals, etc. through search in Photos or Image Search
PRIMARY DATA TYPE Image Data	DATASET CHARACTERISTICS (All numbers are approximate) Total Instances 478k+ Total Classes 6k+ Total Labels 1.27m+ Algorithmically Generated Labels 1.11m+ User Contributed Labels 505k+ Human Verified Labels All labels verified	NATURE OF CONTENT Labeled images of objects (household goods, commercial products), vehicles, plants, animals and people (faces blurred). EXCLUDED DATA All EXIF data including location has been removed PRIVACY PII associated with human subjects removed
DATASET FUNCTION(S) Training Testing	LICENSE TYPE(S) CC-BY-4.0	SUMMARY OF LICENSE PERMISSIONS (CC-BY-4.0) - You are free to share and adapt - Attribution required - You cannot apply any additional restrictions ACCESS COST Open Access
DATA COLLECTION METHOD(S) Crowdsourced	DATA SOURCE(S) - Contributions by global users of the Crowdsourcing app - Vendor data collection efforts	DATA SELECTION All images are opted-in for open-sourcing by Crowdsourcing app contributors
SAMPLING METHOD(S) Unsampled	GEOGRAPHIC DISTRIBUTION 83% India 2% Vietnam 2% Brazil 1% Israel 1% Nigeria 1% Thailand 1% Colombia 1% UAE 8% Others (each less than 1%)	FILTERING CRITERIA - PII: Name tags, Unblurred faces, etc. - Inappropriate Content - Unusable Imagery
LABELING METHOD(S) Human Labels Algorithmic Labels	LABEL TYPE(S) Human Labels Free-form text labels Algorithmic Labels Additional labels LABEL SOURCE(S) Human Labels Image owners Algorithmic Labels Google's internal image annotation algorithm	LABELING PROCEDURE - HUMAN Free-form labels are provided by users of the Crowdsourcing app. The user who has taken the picture provides the label. LABELING PROCEDURE - ALGORITHMIC Labels are resolved against known entity names from Knowledge Graph . Additional labels are added based on Google's internal image annotation system.
VALIDATION METHOD(S) Human Validated	VALIDATION TASK(S) - Human validators verify labels - Human validators flag PII - Human validators filter data VALIDATOR DESCRIPTION(S) Compensated workers based out of India	VALIDATION POLICY SUMMARY Algorithmic and user contributed labels are verified by human validators based out of India. There is a known overlap in algorithmic and user contributed labels. Validators flag any PII content.

⁸ <https://research.google/static/documents/datasets/open-images-extended-crowdsourced.pdf>

Appendix G Trustworthiness Metrics

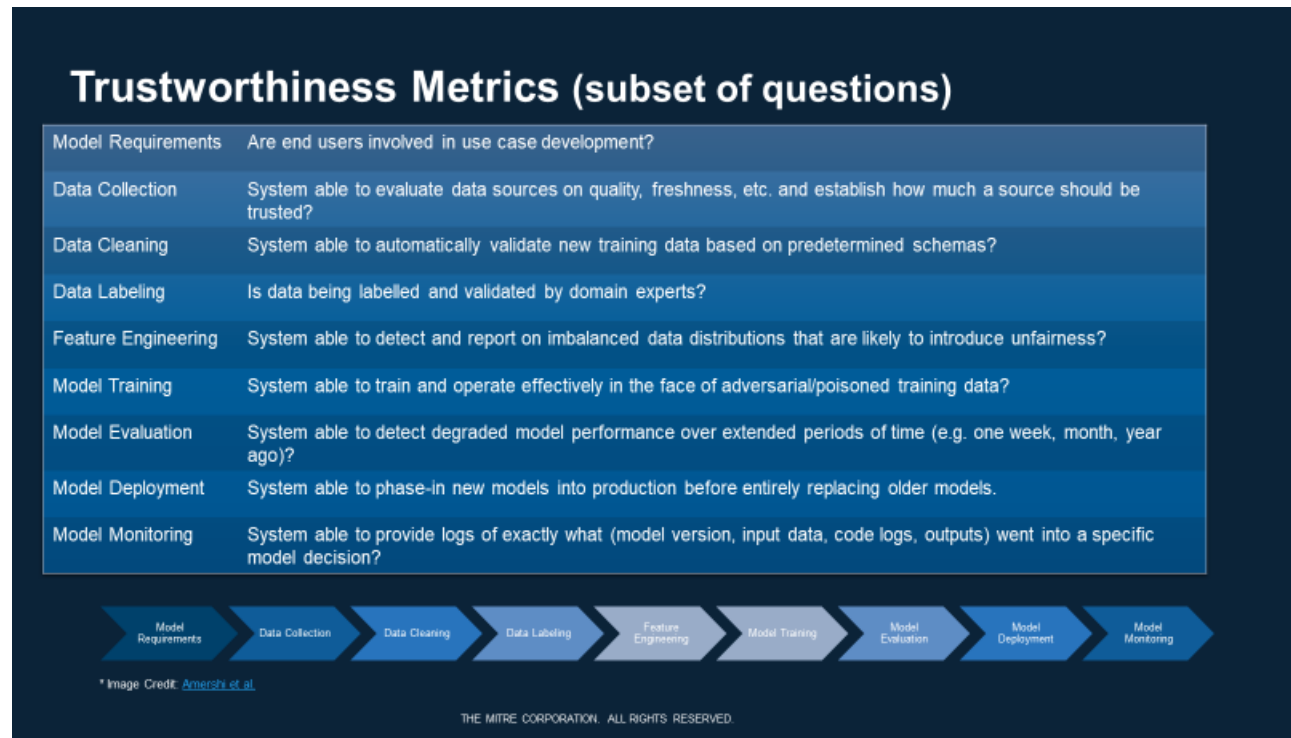


Figure G-1. Trustworthiness Metrics

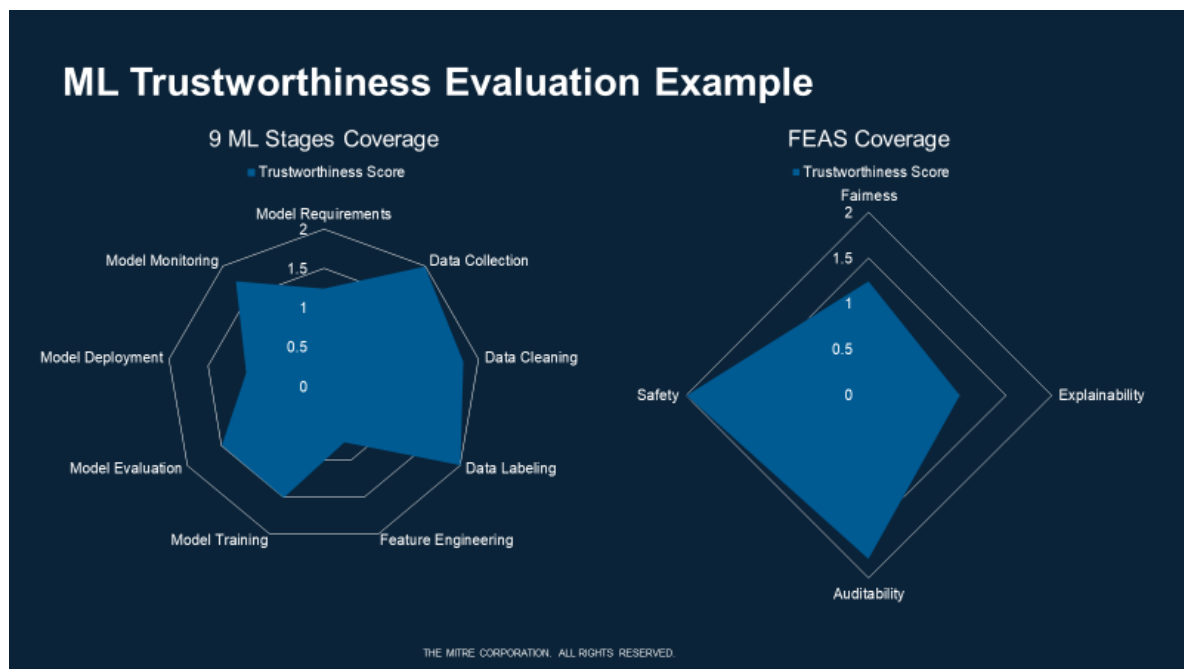


Figure G-2. ML Trustworthiness Evaluation Example

Appendix H Data Split Methods

The goal of data split method is to separate the labeled data into three categories (Train, Validation, and Test) as illustrated in Figure H-1, and each category must possess an equal balance of existing object classes. To achieve a balanced data split, the following are some key considerations for data splitting methods :

- I. Training, validation and test datasets should have samples from the planned object classes described in the requirements document reaching desired (80-10-10) distribution as shown in Figure H-1 (Provided by MORSE Corporation).

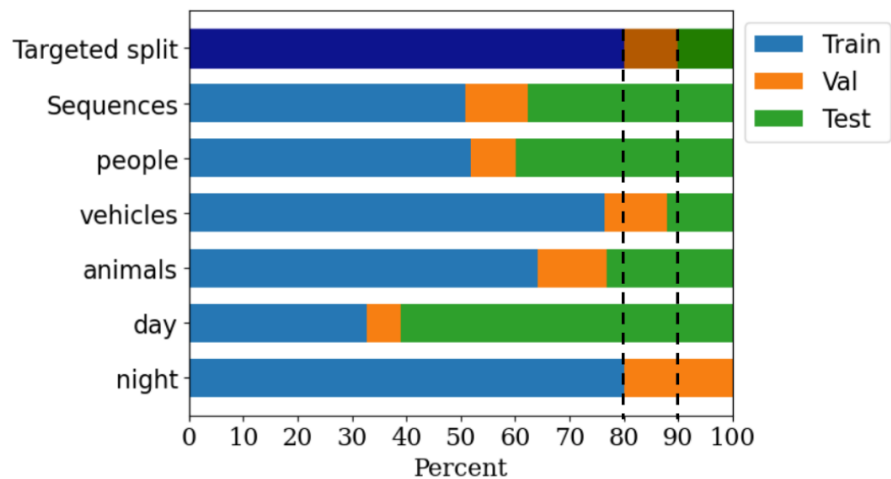


Figure H-1. Data Split Distribution (80-10-10) Example

- II. If the samples lack enough labeled object classes to properly split, consider synthetic labeled data (as Google team's approaches performed for Phase I prototype) to fill in the gaps, OR discuss following options to:
 - **Assign the sample to training**
Pros: Vendor can train on that class immediately
Cons: There is no way to effectively measure the performance of the model on this class. Therefore, any use of the model would be suspect and violate the Department of Defense AI Ethical Principals. For AI to be equitable (e.g., minimize bias), then teams need not only a well-balanced training set; T&E also needs to try to ensure that the "answers" coming from a model can be measured along any relevant metrics.
 - **Assign the sample to test**
Pros: T&E can test on that class immediately.
Cons: There is little point in evaluating a model on attributes that it has not be trained on.

Assign the sample to validation

Pros: Both T&E and model developer can synchronize their testing tools to ensure common results when evaluating the model.

Cons: If an attribute/class is poorly represented over multiple batches, then development teams risk that it will ultimately be overrepresented in the validation set.

- **Reserve for future assignment**

Pros: If an attribute/class is poorly represented over multiple batches, then development and test teams can wait to split on that particular class until sufficient samples are available.

Cons: Some samples will be unused until they are assigned

- III. It is important to include the ground resolution information for analysis purpose, and it should be included in the metadata fields. This ground resolution information potentially already exists (every frame) as part of the flight videos metadata already. If not, a method can be followed to calculate this essential information.

At the end, it is essential to provide a clear definition of ground truth for the model development prior to applying the data for training and validation. If ground truth is not defined in advance, each model vendor will likely use a different process to generate the ground truth, leading to models targeted at different goals, which may be misaligned with government goals or implemented evaluation metrics.

Since an AIES will go through several iterations of T&E this partitioning may need to be extended across several different test events. This may be mitigated by taking slices of data of the test data set or extending the test data set to enable it to vary from test to test.

Appendix I Taxonomy for AI Cybersecurity and Common Threats to ML

One of the first steps to addressing this challenge is coming up with a taxonomy to describe the many different types of attacks and scenarios where they are most likely to be employed. MITRE has developed the MITRE ATLAS™.

MITRE ATLAS™ (Adversarial Threat Landscape for Artificial-Intelligence Systems), is a knowledge base of adversary tactics, techniques, and case studies for machine learning (ML) systems based on real-world observations, demonstrations from ML red teams and security groups, and the state of the possible from academic research. ATLAS is modeled after the [MITRE ATT&CK® framework](#) and its tactics and techniques are complementary to those in ATT&CK.

ATLAS enables researchers to navigate the landscape of [threats to machine learning systems](#). ML is increasingly used across a variety of industries. There are a growing number of vulnerabilities in ML, and its use increases the attack surface of existing systems. We developed ATLAS to raise awareness of these threats and present them in a way familiar to security researchers.

NIST is also developing a robust “[A Taxonomy and Terminology of Adversarial Machine Learning \(nist.gov\).](#)”

This NIST Interagency/Internal Report (NISTIR) is intended as a step toward securing applications of Artificial Intelligence (AI), especially against adversarial manipulations of Machine Learning (ML), by developing a taxonomy and terminology of Adversarial Machine Learning (AML). Although AI also includes various knowledge-based systems, the data-driven approach of ML introduces additional security challenges in training and testing (inference) phases of system operations. AML is concerned with the design of ML algorithms that can resist security challenges, the study of the capabilities of attackers, and the understanding of attack consequences.

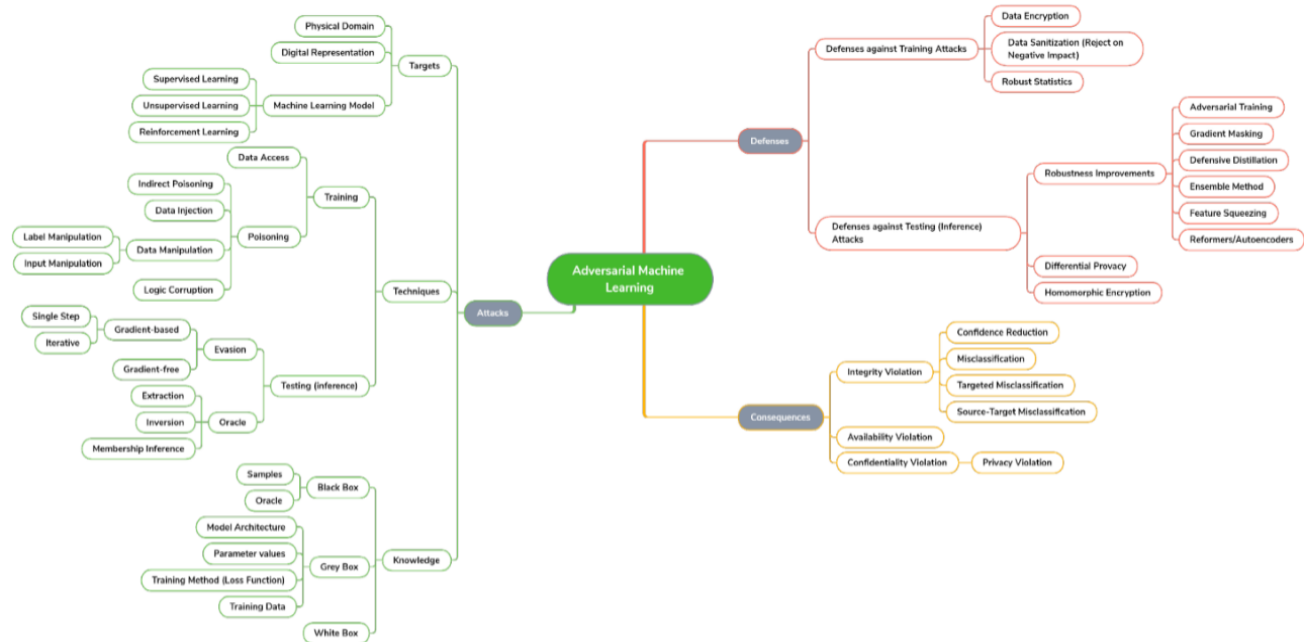


Figure I-1. Adversarial Machine Learning Taxonomy

Common Types of Threats to ML

The types of threats to AI that are cited by ATLAS at the highest level are as follows (detailed technique details can be found on the ATLAS website):

Reconnaissance - The adversary is trying to gather information they can use to plan future operations. Reconnaissance consists of techniques that involve adversaries actively or passively gathering information that can be used to support targeting. Such information may include details of the victim organizations machine learning capabilities and research efforts. This information can be leveraged by the adversary to aid in other phases of the adversary lifecycle, such as using gathered information to obtain relevant ML artifacts, targeting ML capabilities used by the victim, tailoring attacks to the particular models used by the victim, or to drive and lead further Reconnaissance efforts.

Resource Development - The adversary is trying to establish resources they can use to support operations. Resource Development consists of techniques that involve adversaries creating, purchasing, or compromising/stealing resources that can be used to support targeting. Such resources include machine learning artifacts, infrastructure, accounts, or capabilities. These resources can be leveraged by the adversary to aid in other phases of the adversary lifecycle, such as ML Attack Staging.

Initial Access - The adversary is trying to gain access to the system containing machine learning artifacts. The target system could be a network, mobile device, or an edge device such as a sensor platform. The machine learning capabilities used by the system could be local with onboard or cloud enabled ML capabilities. Initial Access consists of techniques that use various entry vectors to gain their initial foothold within the system.

ML Model Access - An adversary is attempting to gain some level of access to a machine learning model. ML Model Access consists of techniques that use various types of access to

the machine learning model that can be used by the adversary to gain information, develop attacks, and as a means to input data to the model. The level of access can range from the full knowledge of the internals of the model to access to the physical environment where data is collected for use in the machine learning model. The adversary may use varying levels of model access during the course of their attack, from staging the attack to impacting the target system.

Execution - The adversary is trying to run malicious code. Execution consists of techniques that result in adversary-controlled code running on a local or remote system. Techniques that run malicious code are often paired with techniques from all other tactics to achieve broader goals, like exploring a network or stealing data. For example, an adversary might use a remote access tool to run a PowerShell script that does Remote System Discovery.

Persistence - The adversary is trying to maintain their foothold. Persistence consists of techniques that adversaries use to keep access to systems across restarts, changed credentials, and other interruptions that could cut off their access. Techniques used for persistence often involve leaving behind modified ML artifacts such as poisoned training data or backdoored ML models.

Defense Evasion - The adversary is trying to avoid being detected by security software. Defense Evasion consists of techniques that adversaries use to avoid detection throughout their compromise. Techniques used for defense evasion include evading ML-enabled security software such as malware detectors.

Discovery - The adversary is trying to figure out your environment. Discovery consists of techniques an adversary may use to gain knowledge about the system and internal network. These techniques help adversaries observe the environment and orient themselves before deciding how to act. They also allow adversaries to explore what they can control and what's around their entry point in order to discover how it could benefit their current objective. Native operating system tools are often used toward this post-compromise information-gathering objective.

Collection - The adversary is trying to gather ML artifacts and other related information relevant to their goal. Collection consists of techniques adversaries may use to gather information and the sources information is collected from that are relevant to following through on the adversary's objectives. Frequently, the next goal after collecting data is to steal (exfiltrate) the ML artifacts, or use the collected information to stage future operations. Common target sources include software repositories, container registries, model repositories, and object stores.

ML Attack Staging - An adversary is leveraging their knowledge of and access to the target system to tailor the attack. ML Attack Staging consists of techniques adversaries use to prepare their attack on the target ML model. Techniques can include training proxy models, poisoning the target model, and crafting adversarial data to feed the target model. Some of these techniques can be performed in an offline manor and are thus difficult to mitigate. These techniques are often used to achieve the adversary's end goal.

Exfiltration - The adversary is trying to steal machine learning artifacts. Exfiltration consists of techniques that adversaries may use to steal data from your network. Data may be stolen for its valuable intellectual property, or for use in staging future operations. Techniques for

getting data out of a target network typically include transferring it over their command-and-control channel or an alternate channel and may also include putting size limits on the transmission.

Impact - The adversary is trying to manipulate, interrupt, erode confidence in, or destroy your systems and data. Impact consists of techniques that adversaries use to disrupt availability or compromise integrity by manipulating business and operational processes. Techniques used for impact can include destroying or tampering with data. In some cases, business processes can look fine, but may have been altered to benefit the adversaries' goals. These techniques might be used by adversaries to follow through on their end goal or to provide cover for a confidentiality breach.

Appendix J Roadmap and Checklists

Advanced planning and a roadmap are essential to perform any T&E for ML and/or AIES. The following is an example of a roadmap/checklist that can be used to engage detailed discussions with project teams on the scope of testing and guide the end-to-end T&E activities involved in the ML T&E.

Table J-1. Sample AI T&E Roadmap

T&E Stage	Description / Sub-levels	Duration (in weeks)	Deliverable
Mission Requirements		8-10	
	Conduct a T&E Workshop and follow up user engagement discussions	2-4	
	Understand Mission-defined task processes, detailed workflows, their importance, and requirements	2	
	Review Mission-defined requirements specifically on MOE/MOP/KPP/KSA to ensure it is testable	2	
	Review Mission-defined requirements specifically on Use Cases with respect Operation Groups/User Communities	2	
	Review CONOPS	1	
	Review and mapping requirements with respect models	1	
	Develop T&E Strategy and go through review cycles Develop Evaluation Card and go through review cycles	6-8	T&E Strategy/ Evaluation Card
Ethics Requirements		2-4	
	Review Ethical considerations and requirements	1	
	Incorporated Ethics Requirements into Data card and Model card process - if applicable	2-4	
	Incorporated Ethics Requirements into T&E Strategy and Specific Test Plan - if applicable	2	
Test Data Planning and Preparation		6	
	Engage Development and Model Team and participate in IPRs	on-going	
	Review Data Cards	on-going	
	Review Model Cards	on-going	
	Access and Evaluate the Data Sets used for Models	2-4	
	Strategize and Coordinate with System/Model teams for Test Data and Validation Data	4	
	- Data Split based on agreed allocation between Training/Validation/Test	2	

T&E Stage	Description / Sub-levels	Duration (in weeks)	Deliverable
	- Routine Data Split with continue data feeds	on-going	
Test Environment and Account(s) Setup		4	
	Accounts	1	
	Coordination of cycle/frequency of Ingest Data and Model Delivery	1	
	Tool Installations	1	
	Configuration and Setup	1	
T&E AI Tool / Model		8-12	
	Work with Model team to confirm and/or define Input/output data/additional requirement, if not available	2	
	Work with Model team to confirm and/or define Algorithm Metrics (accuracy, recall, precision, etc.)	2	
	Develop an Algorithm Test methods and approach for leadership approval	1-2	
	Develop an Algorithm Test Plan to validate planned metrics, if required	2-4	Algorithm Test Plan
	Algorithm Test Execution / Compile Test Metrics from Model Development Team(s)	1	Algorithm Quick Look Report
	Perform Algorithm Results Analysis and Develop Test Report	4-8	Algorithm Test Report
	- Periodic Rerun of T&E Algorithm	on-going	
System Integration with AI Tools		8-12	
	Develop a System Test Plan and scenarios to validate PIRs, mapped to the respect models	4-6	
	Develop a System Test Plan and scenarios to validate MPRD - Use Cases	4-6	
	Develop a System Test Plan and scenarios to validate other relevant specifications within following repeated cycles	4-6	System Test Plan
	- AI Development	1	
	- Security Considerations	1	
	- Ethics Standards	1	
	- Tool V&V specifications	1	
	System Test Execution / Compile a Quick Look Report	2	System Test Quick Look Report

T&E Stage	Description / Sub-levels	Duration (in weeks)	Deliverable
	Compile System Test Results and Develop Test Report	4-6	System Test Report
System Integration with Operation Hardware and Software		4-6	
	Work with involved workgroups to define Reliability, interoperability, and compatibility with operationally relevant hardware and software	2	
	Work with involved workgroups to further define CONOPS and/or KPP/KSA	2	
	Develop Operation Test Plan and scenarios to validate CONOPS and/or MPRD - KPP/KSA for effectiveness and security (cyber and adversarial)	4-6	
	Develop Operation Test Plan to validate Reliability, Interoperability, and Compatibility with operationally relevant hardware and software	4-6	Operation Test Plan
Operational / Human-Machine Integration Test		10-14	
	Perform Human-machine integration in an operationally realistic environment	2	
	- Operation Training and/or Operation in M&S Environment	2	
	- Iteration and Repetition, as needed	2	
	Validate CONOPS in the OT environment	2	
	- Iteration and Repetition, as needed	2	
	Validate Requirements – MOE/MOP/KPP/KSA in the OT environment	2	
	- Iteration and Repetition, as needed	2	
	Validate Reliability, Interoperability, and Compatibility with operationally relevant hardware and software	2	
	- Iteration and Repetition, as needed	2	
	Operation Test Execution / Compile a Quick Look Report	1	Operation Test Quick Look Report
	Compile Operation Test Results and Develop Test Report	4-6	Operation Test Report
User Acceptance Test/Usability Evaluation		6-10	
	Work with Mission Groups/User Community for a UAT - if required	4	UAT Test Results
	Work with Mission Groups/User Community to conduct a Usability Evaluation using SUS	1	Usability Evaluation Questionnaire

T&E Stage	Description / Sub-levels	Duration (in weeks)	Deliverable
	UAT Test Execution / Compile a Quick Look Report	1	UAT Quick Look Report
	Compile UAT Test Results/Survey Results and Develop Test Report	2-4	UAT Test Report and Usability Survey Results
T&E Reporting		4	
	Finalize all Algorithm/System/Operation Test Results and Usability Survey to develop a final T&E Report	4	Final T&E Test Report
	Conduct a final T&E Assessment briefing for leadership	1	T&E Assessment Briefing
T&E Rework Cycles, as needed		4-8	
	Conduct follow up remediation report and/or additional T&E Re-Assessment briefing for leadership	1	T&E Re-Assessment Briefing

The T&E process and activities are always involved with iterations and repetitions. This general T&E roadmap sample is simply meant to provide an overall structure of T&E milestones/governance processes/high level T&E activities and approvals without spelling out all dependencies and details. Similarly, the duration is also a rough estimate as an example, but does not necessarily represent bounding/limitation of needed adjustments or extensions.

Whenever feasible, SMEs and T&E tools should be adopted to provide early preview of algorithm results in early stages, allowing model development teams to immediately assess for model performance and start iterating to improve the model and/or data. With the phased and iterative approach, T&E value will be added early and often in the model development process, instead of a single-event process that holds the model back from being deployed.

The roadmap detailing the test stages/milestones and support planning provides a standardized guide for design, development, customization, and deployment of T&E plans. The checklist also enforces the tracking of time/resource on specific ML test executions, model evaluations, performance assessments, and final test reports/briefings.

Appendix K MLTS and Explanation Satisfaction Scale

Emerging practices are forming to define assessment, in a simple and repeatable manner, of a user's trust in an AIES. A Machine Learning Trust Score (MLTS) was developed to be used during testing in order to survey users' experience and unaddressed needs after they have used an AI. The MLTS is modeled after the System Usability Score, a widely used industry standard measure for traditional software user feedback. SMEs can then analyze these scores to gain specific insights from users' feedback and identify additional improvements for future system upgrades. The MLTS measures users' trust of an ML system based on the dimensions of Ability, Benevolence, Integrity and Predictability (ABIP) and users' perception of Fairness, Explainability, Auditability, and Safety (FEAS). MLTS exposes general areas where user trust is lacking so that issues may be dealt with using targeted investigations.

When this trust score is paired with a survey on the users experience in the domain the AIES tool enables it can also inform on other factors of trust. Over trust is the case where a user may take the recommendations of the AIES in cases where they shouldn't but are unable to discern that they should not trust the AIES and believe or act upon its results. Under trust is demonstrated by a case where a user does not have enough confidence in the AIES system; this can be due to a number of factors such as the consistency of performance of the AIES (right, right, wrong) or the transparency of the information used to derive its decisions. The best situation is when the user has a level of calibrated trust that is equal to the systems reliability.

Machine Learning Trust Score (MLTS) Questions

ID	Question	FEAS / ABIP
1	I feel that I understand where system biases are likely to occur.	Fairness
2	I do not understand what data the system considers for its decisions	Explainability
3	I found that the results were clear, and I could easily explain the rationale to a peer.	Explainability
4	I found that I was not able to sufficiently validate system results within the system.	Auditability
5	I found that I was able to ignore, override, or adjust system decisions when they were wrong.	Safety
6	I do not feel that I understood the range of situations where the system's capabilities are applicable.	Ability
7	I found that the system enhances my ability to perform my job.	Ability
8	I felt that the system was trying to accomplish goals that were different than mine.	Benevolence
9	I felt confident when I made decisions based on system recommendations.	Integrity
10	I found that the system did not produce outputs consistently enough to be predictable.	Predictability

1 - Completely Disagree

2 - Somewhat Disagree

3 - Neutral

4 - Somewhat Agree

5 - Completely Agree

THE MITRE CORPORATION. ALL RIGHTS RESERVED

Figure K-1. Machine Learning Trust Score (MLTS) Questions [64]

Hoffman et al. propose an explanation satisfaction checklist for measuring users' acceptance of model fit on the basis of AI explainability . Items on this checklist address several key expectations of an explainable AI (XAI) capability, including an accessible representation of the modelling process, an understanding of the model's applicability and relevance, and an

interpretation of goodness of fit in terms of accuracy and system confidence scores. Users are then asked to rate each item on a 3-point scale indicating whether they perceive a given explainability factor to be essential, useful but not essential.

Table K-1. Explanation Satisfaction Scale [65]

For each item listed below, please indicate whether you believe that the item is: • Essential for measuring Explanation Satisfaction, • Useful but not Essential, or • Not Necessary for measuring Explanation Satisfaction. Indicate your response by circling the appropriate number.				
		Essential	Useful but not essential	Not Necessary
1.	I understand this explanation of how the [software, algorithm, tool] works.	1	2	3
2.	This explanation of how the [software, algorithm, tool] works is satisfying.	1	2	3
3.	This explanation of how the [software, algorithm, tool] works has sufficient detail.	1	2	3
4.	This explanation of how the [software, algorithm, tool] works contains irrelevant details.	1	2	3
5.	This explanation of how the [software, algorithm, tool] works seems complete.	1	2	3
6.	This explanation of how the [software, algorithm, tool] works tells me how to use it.	1	2	3
7.	This explanation of how the [software, algorithm, tool] works is useful to my goals.	1	2	3
8.	This explanation of says how accurate the [software, algorithm, tool] is.	1	2	3
9.	This explanation lets me judge when I should trust and not trust the [software, algorithm, tool]	1	2	3

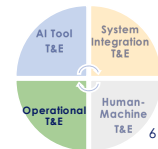
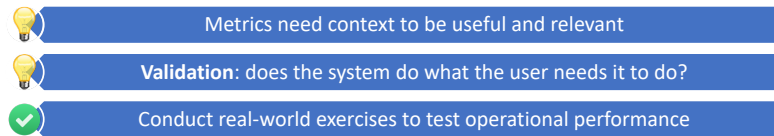
Appendix L CDAO T&E Workshop - Group Exercise

Followings are the workshop group exercise details in the [JAIC \(transitioned to CDAO\) T&E Workshop_20211002.pdf](#).

Group Exercise 1 – Operational Testing and User Engagement

Task: Identify the Operational T&E issues

1. What is this system's value proposition?
2. Is there a CONOPS?
3. Who are the operators? What do they do?
4. Where are the operators located?
5. What is the operational environment?
6. What are the operator's challenges?
7. Who are the other stakeholders? (e.g. people that need to use products of the system, but don't necessarily use it directly themselves.)
8. What is our goal? (e.g. shortening the time of execution of a task, the precision of the task, saving cost,...)
9. What success criteria and relevant operational metrics? (e.g. MOEs, KPPs, MOPs, KSAs)



Group Exercise 1 – Operational Testing and User Engagement

Task: Identify the Operational T&E issues

10. What kind of skills or knowledge are needed to succeed at this task?
11. How often do you do this task?
12. What information is available for you to accomplish the task?
13. What tools are currently used to complete the task?
14. How did you learn how to complete the task?
15. What are the time constraints on the task?
16. Is the task done alone or coordinating with others?
17. How are findings communicated?
18. What happens when things go wrong?
19. Will you be focused on one task or switch between tasks?
20. What type of tool/information do you wish you had to help you be more effective at this task?
21. Is your current process efficient? Do you think there is a better way to complete it?

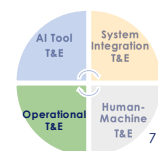
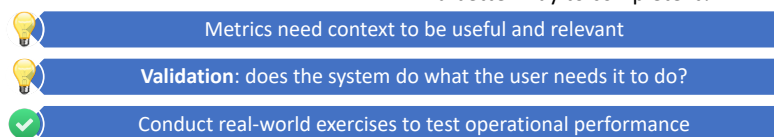


Figure L-1. Group Exercise 1 – Operational Testing and User Engagement

Task: Identify the AI Algorithmic T&E specific testing issues

1. Do we have a specific machine learning component in the system?
2. Are we predicting certain measurements (regression, e.g. what is a price of the house based on the square footage)?
3. Are we forecasting? (Given last 5 months demands, what is the demand for next month?)
4. Are we classifying items (dog vs cat, good situation vs bad situation, etc.)?
5. What does the data look like? Is there an ontology available? Descriptive statistics?
6. How is the data labeled?
7. What are the potential sources of bias?
8. Are there data cards / model cards? What is the process for managing them?

Metrics

Data Split



Data needs to be understood and split into training, shared validation, and protected test sets



Test and quantify model performance along relevant metrics and scenarios with statistically representative test data

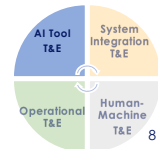


Figure L-2. Group Exercise 2 – AI Tool/Algorithm Testing

Task: Identify the Human - Machine specific testing issues

1. How are these AI/ML tasks being done today by humans?
2. Do we have any relevant metrics on the current process? (time it takes to complete the task, precision of the task, etc.)
3. How can we translate these metrics to proper algorithmic testing metrics?
4. Why might stakeholders not trust this capability? What can be done to give them confidence to trust the capability?



Metrics need context to become useful information



Operators need to be able to trust the system



Qualitative and quantitative assessments



E.g. Survey operators for their feedback, System Usability Scale assessment, etc.

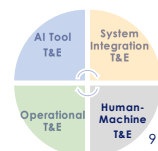


Figure L-3. Group Exercise 3 – Human-Machine Testing

Task: Identify the System Integration T&E specific testing issues

1. Are there system, network, and architecture diagrams available?
2. Where is the development environment? (e.g. AWS Gov Cloud, Vendor specific, etc.)
3. How to perform T&E on Vendor's environment?
4. Where is the deployment environment?
5. How to perform T&E on the deployment environment?
6. Are there cross-domain implications? Do components of the system cross different security or ownership boundaries?
7. What are the performance requirements?
8. What are the integration challenges?

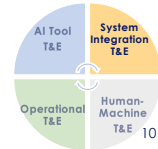
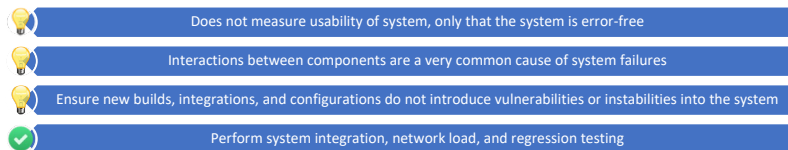
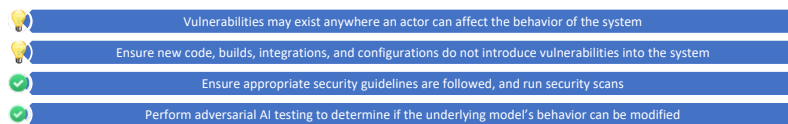


Figure L-4. Group Exercise 4 – System Integration Testing

Task: Identify the Security T&E specific testing issues

1. What are the security threats? (ex. Network connections, Sensors, Data, etc.)
2. Are there network connections?
3. Are there sensors?
4. Are there integrations with other systems?
5. What are the applicable security guidelines? (STIGs?)
6. Are dynamic analysis security tools appropriate?
7. Are static analysis / source code scanning tools appropriate?



Security T&E

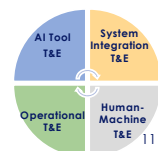


Figure L-5. Group Exercise 5 – Security Testing

Appendix M Acronyms and Abbreviations

TERM	DEFINITION
AA	Adversarial Assessments
ACD	Adversarial Cybersecurity DT&E
AI	Artificial Intelligence
AIES	Artificial Intelligence Enabled Systems
AML	Adversarial Machine Learning
AO	Authorizing Official
API	Application Programming Interface
ATLAS	Advanced Threat Landscape for Artificial-Intelligence Systems
ATO	Authorization to Operate
cATO	Continuous Authorization to Operate
BAA	Broad Agency Announcement
CDAO	Chief Digital and Artificial Intelligence Office
CI/CD	Continuous Integration and Continuous Delivery
CNS	Capability Need Statements
COA	Course of Action
CONOPS	Concept of Operations
COTS	Commercial off the shelf
CPU	Central Processing Unit
CSO	Commercial Solutions Offering
CTR	Contractor
CVI	Cooperative Vulnerability Identification
CVPA	Cooperative Vulnerability and Penetration Assessments
DAU	Defense Acquisition University
DevOps	Development Operations
DoD	Department of Defense
DPI	Dots-per-inch
DRL	Deep Reinforcement Learning
DT&E	Development Test & Evaluation
DTE&A	Developmental Test, Evaluation and Analysis
EDA	Exploratory Data Analysis

TERM	DEFINITION
ETL	Extraction, Transformation, and Loading
GAO	U.S. Government Accountability Office
GFI	Government-Furnished information
GPU	Graphics Processing Unit
HMT	Human-Machine Teaming
IP	Intellectual Property
JADC2	Joint All-Domain Command and Control
JAIC	Joint Air Intelligence Center
JCIDS	Joint Capabilities Integration and Development System
JSON	Java Script Object Notation
KPI	Key Performance Indicator
KPP	Key Performance Parameter
KSA	Key System Attribute
M&S	Modeling and Simulation
MCT	Model Card Toolkit
MDO	Multi-Domain Operations
ML	Machine Learning
MLOps	Machine Learning Operations
MLTS	Machine Learning Trust Score
MOEs	Measures of Effectiveness
MOPs	Measures of Performance
MT	Machine Translation
MTA	Middle Tier of Acquisition
MUC	Message Understanding Conference
NER	Named Entity Recognition
NIST	National Institute of Standards and Technology
NLP	Natural Language Processing or Neuro Linguistic Programming
NN	Neural Network
OCR	Optical Character Recognition
O&M	Operations & Maintenance
OT	Operation Test
OT&E	Operational Test and Evaluation

TERM	DEFINITION
PII/PHI	Personally Identifiable Information/Personal Health Information
PMI	Pointwise Mutual Information
PMO	Program Management Office
QA	Quality Assurance
R&D	Research and Development
RFI	Request for Information
RFP	Request for Proposal
RMF	Risk Management Framework
ROI	Return on Investment
SAP	Software Acquisition Pathway
SELC	Systems Engineering Lifecycle
SME	Subject Matter Expert
SPDX	Software Package Data Exchange
SW	Software
T&E	Test and Evaluation
TALLAL	Tag-a-Little, Learn-a-Little
TBD	To be determined
TEMP	Test and Evaluation Master Plan
TEVV	Testing, Evaluation, Validation & Verification
TPU	Tensor Processing Unit
TS	Test Strategy
URL	Unified Markup Language
V&V	Verification and Validation
VOLT	Validated Online Lifecycle Threat
XAI	Explainable Artificial Intelligence

Bibliography