Carnegie Mellon University
Software Engineering Institute

Modeling to Support DoD Acquisition Lifecycle Events

Version 1.4

Julie Cohen
Tom Merendino
Rob Wojcik

April 2022

SPECIAL REPORT
CMU/SEI-2021-SR-034

http://www.sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense and the US Army Development Command Aviation and Missile Center (DEVCOM AvMC) under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM21-0767

Table of Contents

List of Tables

List of Figures

1 Introduction

The intent of this document is to provide suggestions for producing requirement, system, and software models that will be used to support various DoD system acquisition lifecycle events including System Requirements Reviews (SRR), System Functional Reviews (SFR), Software Specification Reviews (SSR), Preliminary Design Reviews (PDR), and Critical Design Reviews (CDR). The models will be used to provide information about the systems and/or software being reviewed at each event that is needed to satisfy the stated purposes of those events.

At present, the document only covers modeling suggestions for SRRs and SFRs. Future work will focus on providing modeling suggestions for SSR, PDR, and CDR.

Important Note:

The definitions for some of the terms used in this document may differ from the reader's personal definitions and "commonly accepted" and "standard" definitions found on the internet and in a broad range of documents. For example, the definitions for "Government Requirement" and "System Requirement" as used in this document are as follows:

Government Requirement	Describes a condition or capability needed by a stakeholder to solve a problem or achieve an objective. A condition or capability that must be met or possessed by a solution or solution component to satisfy a contract, standard, specification, or other formally imposed documents. – ("Requirement", Wikipedia.org) In practice, such requirements come from many sources including informal conversations up through formal documents (e.g., Capability Development Documents, Regulations, and Specifications). Contractors are expected to take such requirements and translate them into System Requirements (i.e., requirements for a system that will satisfy the Government's requirements). The term "Government Requirements" as used in this document refers to any such requirements put forth by the Government prior to those requirements being translated into System Requirements.
System Requirement	Describes a functional, behavioral, or quality attribute characteristic a system must satisfy to ensure the system meets an organization's business, technical, and/or operational requirements. Such requirements are the result of translating Government Requirements into requirements for a system that will satisfy those requirements. By SRR, system requirements should be documented to the point where reviewers can determine whether or not contractors have understood and properly translated Government Requirements into system requirements. By SRR, there is NOT an expectation that system requirements have been decomposed to a low level of abstraction. By SFR, system requirements should be decomposed to a level that provides a satisfactory basis for preliminary design activities. The term "System Requirement" as used in this document refers to requirements that result from translating requirements put forth by the Government into requirements that a system must meet to satisfy the Government's requirements.

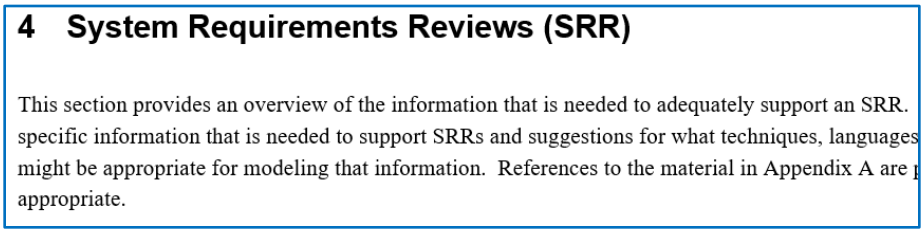
We use this approach of defining exactly what we mean by the terms used in the document rather than relying on readers knowing or understanding "commonly understood" or "standard" definitions for terms that often overloaded, unclear, conflicting, misused, etc. To this end, we have provided a "Concepts and Glossary" section as well as footnotes throughout the document to familiarize readers with those terms.

2 How to Use This Document

1. Become familiar with the concepts and terms used in the document by reviewing Section 3. Please remember, definitions for some of the terms used in this document may differ from the reader's personal definitions and "commonly accepted" and "standard" definitions found on the internet and in a broad range of documents.
2. Decide which lifecycle event your models will support and review the section of the document that describes the event. As an example, the following describes how to review the section for the SRR event. Note that screen captures are provided as visual aids to assist the reader in understanding each step and in locating the document sections related to each step and DO NOT contain the full text of those sections.

   a. Read the introduction to the section as shown in Figure 1:

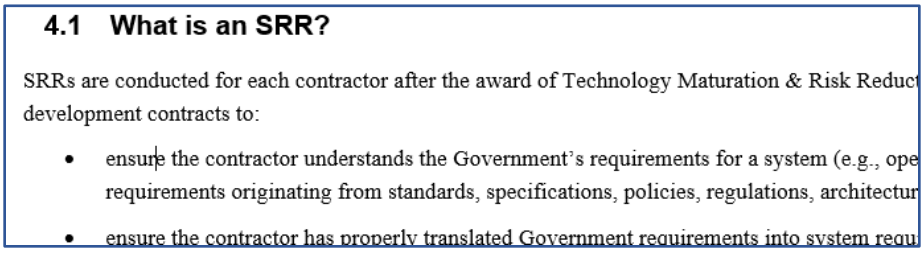
   Figure 1 – Example: Read Introduction for Lifecycle Event of Interest

   
   4 System Requirements Reviews (SRR)

   This section provides an overview of the information that is needed to adequately support an SRR. specific information that is needed to support SRRs and suggestions for what techniques, languages might be appropriate for modeling that information. References to the material in Appendix A are p appropriate.

   b. Read the description and purpose of the event as shown in Figure 2:

   Figure 2 – Example: Read the Lifecycle Event Description and Purpose

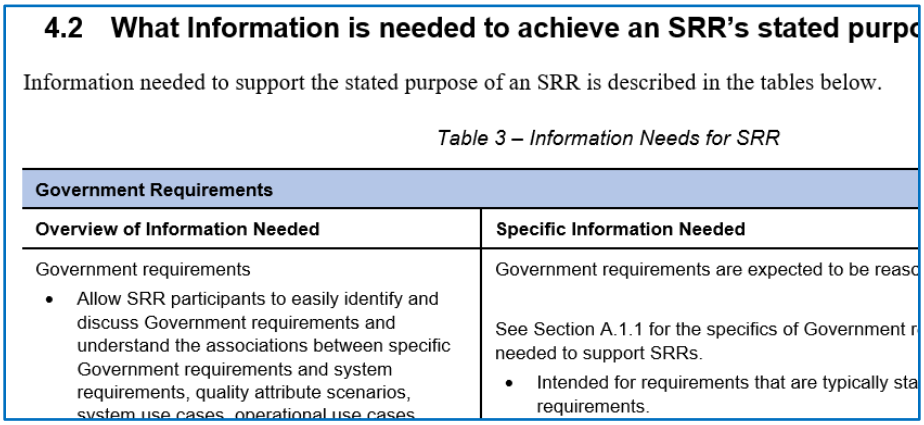
   
   4.1 What is an SRR?

   SRRs are conducted for each contractor after the award of Technology Maturation & Risk Reduct development contracts to:

   - ensure the contractor understands the Government's requirements for a system (e.g., ope requirements originating from standards, specifications, policies, regulations, architectur
   - ensure the contractor has properly translated Government requirements into system requ

   c. Review the table in the "What information is needed…" section as shown in Figure 3.

   Figure 3 – Example: Identify Information Needed for Lifecycle Event

   
   4.2 What Information is needed to achieve an SRR's stated purpo

   Information needed to support the stated purpose of an SRR is described in the tables below.

   Table 3 – Information Needs for SRR

Government Requirements	
Overview of Information Needed	**Specific Information Needed**
Government requirements • Allow SRR participants to easily identify and discuss Government requirements and understand the associations between specific Government requirements and system requirements, quality attribute scenarios, system use cases, operational use cases	Government requirements are expected to be reasc See Section A.1.1 for the specifics of Government r needed to support SRRs. • Intended for requirements that are typically sta requirements.

   In Figure 3, the left-hand column in Table 3, "Overview of Information Needed", describes the types of information needed to support the event and why that information is needed.

The right-hand column Table 3, "Specific Information Needed", directs the reader to various sections in Appendix A that provide the details regarding what specific information is needed. In the above example, the reader is directed to Section A.1.1 to see what specific information is needed regarding Government Requirements.

3. As directed by the table in the previous step, go to the section in Appendix A that describes what specific information is needed. Continuing with the above example, as shown in Figure 4, Section A.1.1 provides what specific information is needed for Government Requirements:

Figure 4 – Example: Identify Specific Information Needed for Lifecycle Event

A.1 Requirement Information

A.1.1 Essential Requirement Information

Specific information about government requirements, system requirements, and various associati
achieving the stated purposes of SRRs and SFRs is provided in Table 9. Information in the table
"SFR" in the "Essential" column is considered essential for every requirement at the correspondi
the table marked '-' in the "Essential" column may be useful but is not considered essential for S
marked as essential for SSR, PDR, and/or CDR in subsequent revisions of this document).

Table 9 - Essential Requirement Information

Information Needed	Essential	Description
Requirement ID	SRR, SFR	Unique identifier (e.g., requirement number).
Requirement description	SRR, SFR	Textual description of the requirement.
Requirement rationale	SRR, SFR	Provides a rationale for the requirement (e.g., th system constraint like "The C++ programming la software development." might be "All of our dev program in C++. Choosing any other programm increase cost and schedule").
Requirement categories	_	One or more logical requirement categories to w For example, the requirement "The life preserve

a. In Figure 4, the left-hand column of Table 9, "Information Needed", shows that "Requirement ID", "Requirement description", "Requirement rationale", "Requirement categories", etc. are needed for each requirement.

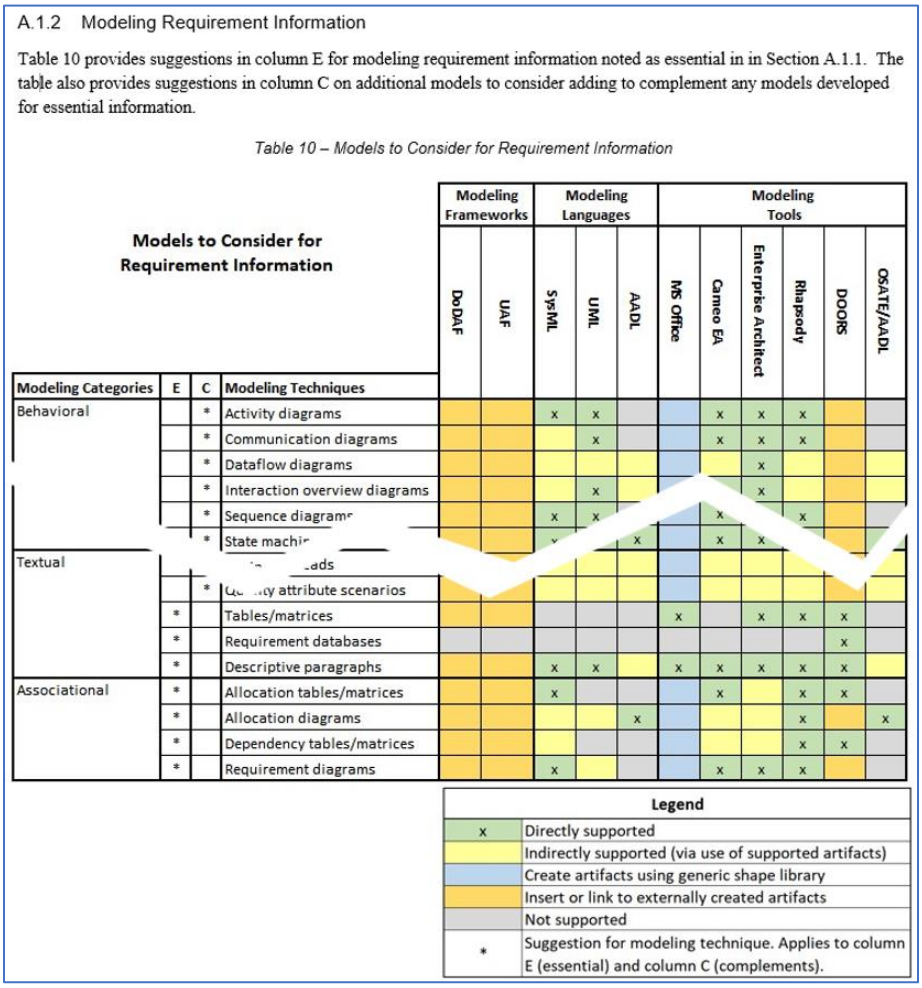
b. The middle column of Table 9, "Essential", indicates whether or not the information is considered essential and for which event(s) the information is essential. In this example, Table 9 indicates that "Requirement ID", "Requirement description", and "Requirement rationale" are considered to be essential for each Government Requirement for both SRR and SFR. "Requirement categories" is not considered to be essential.

c. The right-hand column of Table 9, "Description", describes each information item.

4. Next, review the section in Appendix A that provides suggestions for modeling the information identified in Step 3. Continuing with the above example, Figure 5 shows the table from Section A.1.2 that provides suggestions for modeling techniques, frameworks, languages, and tools that can be used to model requirement information:

    a. First review the table in Section A.1.2, specifically column E (which stands for "essential"), for suggestions regarding what modeling techniques to use for modeling the essential requirement information identified in Step 3. The rows marked with an asterisk in column E are suggested modeling techniques. Continuing with the above example, "Tables/matrices", "Requirement databases", "Descriptive paragraphs", "Allocation tables/matrices", etc. are suggested modeling techniques for essential requirement information.

Figure 5 – Example: Identify Modeling Techniques



A.1.2 Modeling Requirement Information

Table 10 provides suggestions in column E for modeling requirement information noted as essential in in Section A.1.1. The table also provides suggestions in column C on additional models to consider adding to complement any models developed for essential information.

Table 10 – Models to Consider for Requirement Information

Models to Consider for Requirement Information				Modeling Frameworks		Modeling Languages			Modeling Tools					
				DoDAF	UAF	SysML	UML	AADL	MS Office	Cameo EA	Enterprise Architect	Rhapsody	DOORS	OSATE/AADL
Modeling Categories	**E**	**C**	**Modeling Techniques**											
Behavioral		*	Activity diagrams			x	x			x	x	x		
		*	Communication diagrams				x			x	x	x		
		*	Dataflow diagrams								x			
		*	Interaction overview diagrams				x				x			
		*	Sequence diagrams			x	x			x		x		
		*	State machines					x		x	x			
Textual			Use cases											
		*	Quality attribute scenarios											
	*		Tables/matrices						x		x	x	x	
	*		Requirement databases										x	
	*		Descriptive paragraphs			x	x		x	x	x	x	x	
Associational	*		Allocation tables/matrices			x				x		x	x	
	*		Allocation diagrams					x				x		x
	*		Dependency tables/matrices									x	x	
	*		Requirement diagrams			x				x	x	x		

Legend	
x	Directly supported
	Indirectly supported (via use of supported artifacts)
	Create artifacts using generic shape library
	Insert or link to externally created artifacts
	Not supported
*	Suggestion for modeling technique. Applies to column E (essential) and column C (complements).

    b. Next review the table in Section A.1.2, specifically column C (which stands for "complements"), for suggestions regarding additional models to consider adding to complement any models developed for essential information. The rows marked with an asterisk in column C are suggested modeling techniques. Continuing with the above example, "Activity diagrams", "Communication diagrams", "Dataflow diagrams", etc. are suggested modeling techniques to use for developing models to complement models developed for essential information.

    c. Finally, review the Modeling Frameworks, Modeling Languages, and Modeling Tools columns in the table for suggestions on frameworks, languages, and tools to use for modeling. Continuing with the above example, the table shows that DoDAF, UAF, SysML, MS Office, Cameo EA, Enterprise Architect. Rhapsody, and DOORS are suggestions for applying the "Allocation tables/matrices" modeling technique.

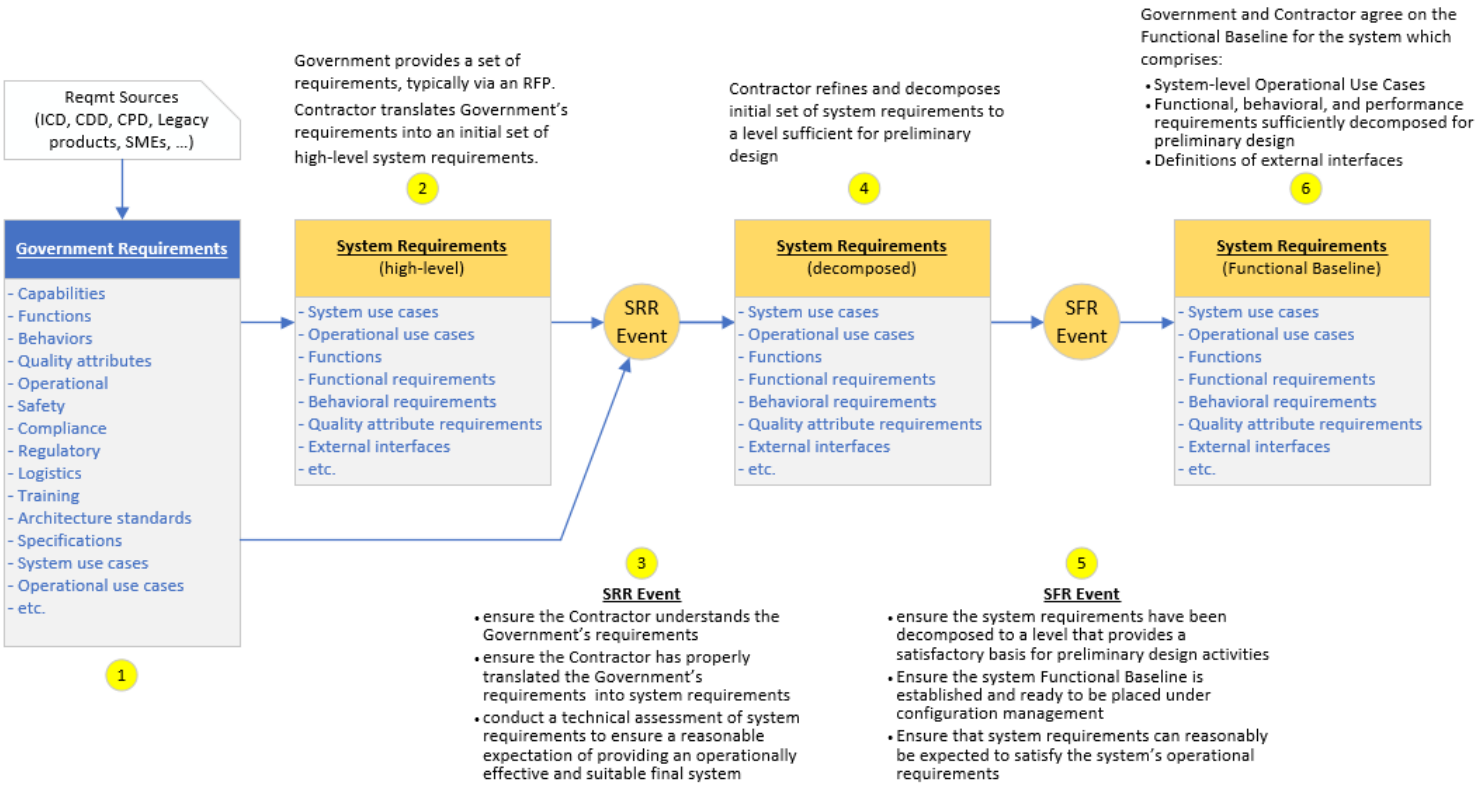
3 Concepts and Glossary

The purpose of this section is to provide an overview of and definitions for concepts and terms that are frequently used within this document to define exactly what we mean by those terms rather than relying on readers knowing or understanding "commonly understood" or "standard" definitions for terms that often overloaded, unclear, conflicting, misused, etc.

3.1 Development Lifecycle – From Government Requirements through SFR

Figure 6 provides an overview of how we picture one segment of the development lifecycle, from the time Government requirements are issued through the conclusion of a System Functional Review (SFR), and what information we think is needed (and eventually modeled) to adequately support the stated purposes of System Requirements Review (SRR) and SFR.

Figure 6 – Overview of Information Needed for SRR and SFR



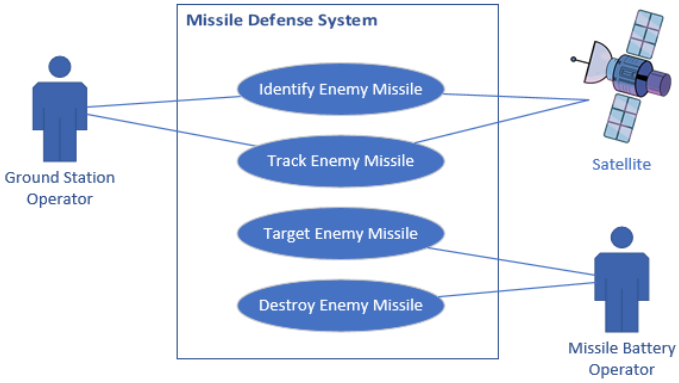
Steps	Descriptions
1	The Government needs to a new system or needs to modify an existing system. To address the need, the Government develops a set of requirements that describe conditions the system must meet and capabilities the system must provide. The requirements are based on various stakeholder needs and organizational objectives and typically address a wide variety of requirement types (e.g., functional, behavioral, quality attribute, regulatory, safety, training, logistics, etc.). The requirements can come from many sources which include anything from informal conversations to formal documents (e.g., Capability Development Documents, Regulations, Architecture Standards, Specifications, etc.). These requirements are referred to as "Government Requirements" in this document.
2	Government Requirements for a system are provided by the Government to a contractor, typically via an RFP. The contractor takes those requirements and begins translating them into system requirements. The contractor's first pass at translating the requirements results in an initial set of high-level system requirements. Requirements that result from translating government requirement into system requirements are referred to as "System Requirements" in this document.

3	The Government conducts a System Requirements Review (SRR). Both Government Requirements and System Requirements are key inputs into the SRR. The contractor is expected to provide a set of high-level System Requirements as input to the SRR. The contractor is NOT expected to have the System Requirements decomposed to a low -level of abstraction by SRR. During the SRR, the Government ensures that the contractor understands the Government Requirements, ensures the contractor has properly translated the Government Requirements into System Requirements, and conducts a technical review of the System Requirements to ensure a reasonable expectation that the System Requirements will result in an operationally effective and suitable final system.
4	After the SRR, the contractor prepares for a System Functional Review (SFR). Prior to the SFR, the contractor is expected to make any necessary adjustments to the high-level System Requirements that were reviewed at the SRR (i.e., refine the high-level requirements) and to decompose the high-level System Requirements to a level sufficient for preliminary design.
5	The Government conducts an SFR. Both Government Requirements and System Requirements are key inputs into the SFR. The contractor is expected to provide a set of System Requirements that includes the high-level requirements that were reviewed at SRR and a decomposition of those requirements to a level that provides a satisfactory basis for preliminary design During the SFR, the Government ensures that the System Requirements developed by the contractor have been decomposed to a level that provides a satisfactory basis for preliminary design activities, ensures that a system functional baseline has been established and is ready to be placed under configuration management, and ensures the system can reasonably be expected to satisfy the system's operational requirements.
6	At the conclusion of the SFR, the Government and Contractor agree on the functional baseline for the system. The functional baseline includes system-level operational use cases, Functional, behavioral, performance requirements sufficiently decomposed for preliminary design, and definitions of external interfaces.

3.2 System Use Cases

Figure 7 provides an overview of system use cases. Our intent is to point out that a system use case shows one possible use of a system and that it is one of many such use cases. From the figure, one can see which actors and use cases the notional missile defense system must support, but there is nothing in the figure that provides any particular context or order in which those use cases will be employed. This brief overview sets the stage for defining what we mean by "operational use case".

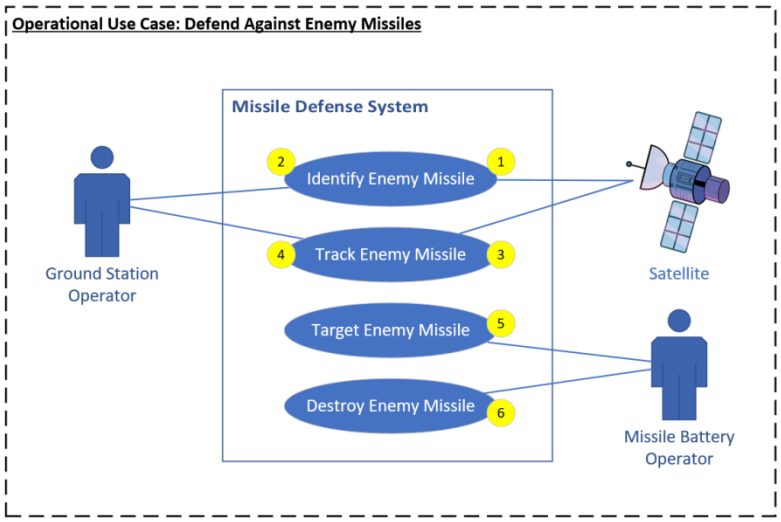
Figure 7 - System Use Cases



3.3 Operational Use Cases

Figure 8 provides an overview of an operational use case. Operational use cases show "day in the life" scenarios that involve multiple actors who employ various system use cases, in a specific order, to achieve a particular objective. In this particular example, the objective is to "defend against enemy missiles" and the operational use case with the same name shows which system use cases are employed, by which actors, and in what order to achieve that objective.

Figure 8 - Operational Use Case



3.4 Glossary

Table 1 provides a glossary for key concepts/terms that are used in this document.

Table 1 – Glossary of Terms and Concepts

Concepts	Descriptions
Functional Baseline	Describes the system's performance (functional, interoperability and interface characteristics) and the verification required to demonstrate the achievement of those specified characteristics. It is directly traceable to the operational requirements contained in the Initial Capabilities Document (ICD). – ("Configuration Management", Systems Engineering Brainbook, DAU.edu) A functional baseline for a system comprises the following: • system use cases • operational use cases • system functions • functional, behavioral, and quality attribute requirements decomposed to a level that provides a satisfactory basis for preliminary design activities • definitions for external interfaces Alternative Definition(s): 1. In configuration management, the initial approved technical documentation for a configuration item. 2. The initial configuration established at the end of the requirements definition phase. - (ISO/IEC/IEEE 24765)
Functional Requirement	A requirement that specifies a function that a system or system component must be able to perform. - (ISO/IEC/IEEE 24765) Alternative Definition(s): A statement that identifies what a product or process must accomplish to produce required behavior and/or results. – (IEEE Standard 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process. 3.1.16)
Government Requirement	Describes a condition or capability needed by a stakeholder to solve a problem or achieve an objective. A condition or capability that must be met or possessed by a solution or solution component to satisfy a contract, standard, specification, or other formally imposed documents. – ("Requirement", Wikipedia.org) In practice, such requirements come from many sources including informal conversations up through formal documents (e.g., Capability Development Documents, Regulations, and Specifications). Contractors are expected to take such requirements and translate them into System Requirements (i.e., requirements for a system that will satisfy the Government's requirements). The term "Government Requirements" as used in this document refers to any such requirements put forth by the Government prior to those requirements being translated into System Requirements.
Operational Requirement	Capabilities, performance measurements (Measures of Effectiveness, Measures of Performance, Measures of Suitability, and Technical Performance Measurements) and processes needed to address mission area deficiencies, evolving threats, emerging technologies, or weapon system cost improvements. They form the foundation for weapon system technical specifications, contract requirements and guide product development so that solutions specifications actively solve the stated problems. Operational requirements focus on how the system will be operated by the users, including interfaces and interoperability with other systems. The operational requirements establish how well and under what conditions the system must perform. Operational requirements are specified in a Capabilities Development Document (CDD). – (Acqnotes.com)
Operational Use Case	Shows a "day-in-the-life" scenario for a system. A typical operational use case might show how and in what order one or more actors interact with a system via specific System Use Cases during the course of a day to achieve various objectives. For example, the operational use case in Figure 8 shows how the various actors interact with the system using various system use cases to achieve the objective "defend against enemy missiles".

Performance Requirement	A statement of the extent to which a function must be executed, generally measured in terms such as quantity, accuracy, coverage, timeliness, or readiness. Alternative Definition(s): 1. The measurable criterion that identifies a quality attribute of a function or how well a functional requirement must be accomplished. – (IEEE Standard 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.3.1.26.) 2. A system or software requirement specifying a performance characteristic that a system/software system or system/software component must possess 3. a requirement that imposes conditions on a functional requirement - (ISO/IEC/IEEE 24765)
Quality Attribute	A measurable or observable property of a system that is used to indicate how well the system satisfies the needs of stakeholders.
Quality Attribute Requirement	Defines a quality attribute a system must have to satisfy a contract, standard, specification, or other formally imposed document.
Quality Attribute Scenario	A technique for specifying a quality attribute requirement for a system. A quality attribute scenario is a brief description of how a system is required to respond to some stimulus. Such scenarios serve as architectural test cases that provide insights into the qualities an architecture supports and any risks associated with fulfilling those scenarios.
System Function	A specification of behavior that describes what a system does between specific system inputs and outputs.
System Functional Requirement	A task (sometimes called action or activity) that must be accomplished when using the system to provide an operational capability (or to satisfy an operational requirement). A Functional Requirement "relates directly to a process the system has to perform as a part of supporting a user task and/or information it needs to provide as the user is performing a task." (Ref "System Analysis and Design, Fifth Edition", Alan Dennis)
System Performance Requirement	1. Measurable criterion that identifies a quality attribute of a function or how well a functional requirement must be accomplished. IEEE Standard 1220-2005 IEEE Standard for the Application and Management of the Systems Engineering Process.3.1.26. 2. A system or software requirement specifying a performance characteristic that a system/software system or system/software component must possess 3. a requirement that imposes conditions on a functional requirement (ISO/IEC/IEEE 24765) System performance requirements are derived from a Capabilities Development Document (CDD) (and possibly other sources… e.g., like for safety, security, and environment requirements). System requirements are documented in a System Requirements Specification.
System Requirement	Describes a functional, behavioral, or quality attribute characteristic a system must satisfy to ensure the system meets an organization's business, technical, and/or operational requirements. Such requirements are the result of translating Government Requirements into requirements for a system that will satisfy those requirements. By SRR, system requirements should be documented to the point where reviewers can determine whether or not contractors have understood and properly translated Government Requirements into system requirements. By SRR, there is NOT an expectation that system requirements have been decomposed to a low level of abstraction. By SFR, system requirements should be decomposed to a level that provides a satisfactory basis for preliminary design activities. The term "System Requirement" as used in this document refers to requirements that result from translating requirements put forth by the Government into requirements that a system must meet to satisfy the Government's requirements.

System Use Case	Describes one possible use of a system from the perspective of an actor (i.e., someone or something) that interacts with the system for a particular purpose. A use case description comprises a set of interactions that describe the "back and forth" between an actor and the system that takes place during the use case (e.g., the actor does A, the system responds with B, the actor does C, the system responds with D, etc.). Figure 7 shows the following actors and use cases:
	• Actors
	  ○ Ground Station Operator
	  ○ Satellite
	  ○ Missile Batter Operator
	• Use Cases
	  ○ Identify Enemy Missile
	  ○ Track Enemy Missile
	  ○ Target Enemy Missile
	  ○ Destroy Enemy Missile

3.5 Acronyms

Table 2 provides a list of acronyms that are used in this document.

Table 2 - Acronyms

Acronyms	Definitions
AADL	Architecture Analysis and Design Language
CDR	Critical Design Review
DoDAF	Department of Defense Architecture Framework
DOORS	Dynamic Object-Oriented Requirements System
MS Office	Microsoft Office
OSATE	Open Source AADL Tools Environment
PDR	Preliminary Design Review
SFR	System Functional Review
SRR	System Requirements Review
SSR	Software Specification Review
SysML	Systems Modeling Language
UAF	Unified Architecture Framework
UML	Unified Modeling Language

4 System Requirements Reviews (SRR)

This section provides an overview of the information that is needed to adequately support an SRR. Appendix A includes specific information that is needed to support SRRs and suggestions for what techniques, languages/frameworks, and tools might be appropriate for modeling that information. References to the material in Appendix A are provided in this section as appropriate.

4.1 What is an SRR?

SRRs are conducted for each contractor after the award of Technology Maturation & Risk Reduction (TMRR) Phase development contracts to:

- ensure the contractor understands the Government's requirements for a system (e.g., operational requirements, requirements originating from standards, specifications, policies, regulations, architectural constraints, etc.)
- ensure the contractor has properly translated Government requirements into system requirements
- conduct a technical assessment of system requirements to ensure a reasonable expectation of providing an operationally effective and suitable system

To these ends, at a high level of abstraction, the information needed during an SRR includes information about:

- Government requirements
- System use cases
- Operational use cases
- System functions
- System requirements
- External interfaces
- Various associations between the above items (e.g., associations between Government requirements and system requirements, associations between quality attribute scenarios, use cases, system functions, and system requirements)
- Various associations between the above items and other artifacts (e.g., associations between requirements and policies, regulations, architectures, and/or standards, associations between requirements and models that clarify those requirements)

4.2 What Information is needed to achieve an SRR's stated purpose?

Information needed to support the stated purpose of an SRR is described in the tables below.

Table 3 – Information Needed for SRR

Government Requirements	
Overview of Information Needed	**Specific Information Needed**
Government requirements • Allow SRR participants to easily identify and discuss Government requirements and understand the associations between specific Government requirements and system requirements, quality attribute scenarios, system use cases, operational use cases, system functions, and other Government requirements.	Government requirements are expected to be reasonably complete prior to SRR. See Section A.1.1 for the specifics of Government requirements information needed to support SRRs. • Intended for requirements that are typically stated as shall, will, or should requirements. • Requirements of this type include capability, functional, behavioral, quality attribute, operational, safety, compliance, regulatory, logistic, training, etc. requirements that the government expects any proposed solutions to meet.
System Requirements	
Overview of Information Needed	**Specific Information Needed**
System use cases • Allow SRR participants to easily identify and discuss system use cases and understand the associations between specific system use cases and Government requirements, system requirements, quality attribute scenarios, operational use cases, system functions, and other system use cases. • Allow SRR participants to visualize typical interactions between specific actors and the system in support of operational use cases.	At SRR, contractors are expected to identify and describe the primary system use cases that will be supported by the system. Refinements to those use cases (e.g., based on discussions during the SRR) and decomposition of those use cases (e.g., into use case extensions) is not expected until SFR. See Section A.2.1 for the specifics of system use case information needed to support SRRs.
Operational use cases • Allow SRR participants to easily identify and discuss operational use cases and understand the associations between specific operational use cases and Government requirements, system requirements, quality attribute scenarios, system use cases, system functions, and other operational use cases. • Allow SRR participants to visualize typical day-in-the-life scenarios where system use cases are applied by one or more actors in various combinations to achieve operational goals/objectives.	At SRR, contractors are expected to identify and describe the primary operational use cases that will be supported by the system. Refinements to those use cases (e.g., based on discussions during the SRR) and decomposition of those use cases (e.g., into use case extensions) is not expected until SFR. See Section A.2.1 for the specifics of operational use case information needed to support SRRs. • Intended for operational use cases that describe "a day in the life" of a system.

System functions • Allow SRR participants to easily identify and discuss functions that will be provided by the system and understand the associations between specific system functions and Government requirements, system requirements, quality attribute scenarios, system use cases, operational use cases, and other system functions.	At SRR, contractors are expected to identify and describe the primary system functions that will be provided by the system. Refinements to those functions (e.g., based on discussions during the SRR) and decomposition of those functions into sub-functions at the lowest level of abstraction is not expected until SFR. See Section A.3.1 for the specifics of system function information needed to support SRRs.
System requirements • Allow SRR participants to easily identify and discuss system requirements and understand the associations between specific system requirements and Government requirements, quality attribute scenarios, system use cases, operational use cases, system functions, and other system requirements.	At SRR, contractors are expected to identify and describe high-level system requirements. Refinements to those requirements (e.g., based on discussions during the SRR) and decomposition of those requirements into requirements at the lowest level of abstraction is not expected until SFR. See Section A.1.2 for the specifics of system requirements information needed to support SRRs. • Intended for requirements that are typically stated as shall, will, or should requirements • Requirements of this type include functional, behavioral, and quality attribute requirements.
External interfaces • Allow SRR participants to easily identify and discuss requirements for external interfaces between the system being developed and external systems.	At SRR, contractors are expected to identify and describe interfaces between the system and external systems. Identification and description of internal system interfaces is not expected until PDR. See Section A.4.1 for the specifics of external interface information needed to support SRRs.

5 System Functional Reviews (SFR)

This section provides an overview of the information that is needed to adequately support an SFR. Specific information that is needed to support SFRs and suggestions for what techniques, languages/frameworks, and tools might be appropriate for modeling that information is outlined in Appendix A. References to the material in Appendix A are provided in this section as appropriate.

5.1 What is an SFR?

SFRs are conducted for each contractor during the Technology Maturation & Risk Reduction (TMRR) Phase of a program to:

- ensure a system's functional, behavioral, and quality attribute requirements have been decomposed to their lowest level and will provide a satisfactory basis for preliminary design activities
- ensure that system requirements can reasonably be expected to satisfy the government's requirements
- ensure that interfaces between the system being developed and external systems are defined and designated as key interfaces
- ensure that a sufficient number of operational use cases for the system have been identified/finalized
- establish the functional baseline for a system

A functional baseline for a system comprises the following:

- system use cases
- operational use cases
- system functions
- quality attribute scenarios
- functional, behavioral, and quality attribute requirements decomposed to a level that provides a satisfactory basis for preliminary design activities
- definitions for external interfaces

To these ends, at a high level of abstraction, the essential information needed during an SFR includes information about:

- system use cases decomposed to their lowest level of abstraction
- operational use cases decomposed to their lowest level of abstraction
- system requirements decomposed to their lowest level of abstraction
- definitions for external interfaces
- associations between requirements (e.g., between operational requirements and system requirements, operational requirements and operational requirements, system requirements and system requirements, operational use cases and system requirements, etc.))
- Various associations between the above items (e.g., associations between Government requirements and system requirements, associations between use cases, system functions, and system requirements)
- Various associations between the above items and other artifacts (e.g., associations between requirements and policies, regulations and/or standards, associations between requirements and models that clarify requirements those requirements)

5.2 What Information is needed to achieve an SFR's stated purpose?

Information needed to support the stated purpose of an SFR is described in the tables below.

Table 4 – Information Needed for SFR

Government Requirements	
Overview of Information Needed	**Specific Information Needed**
Government requirements • Allow SFR participants to easily identify and discuss Government requirements and understand the associations between specific Government requirements and system requirements, quality attribute scenarios, system use cases, operational use cases, system functions, and other Government requirements.	Government requirements are expected to be reasonably complete prior to SRR. Refinements to Government requirements may occur between SRR and SFR. See Section A.1.1 for the specifics of government requirements information needed to support SFRs. • Intended for requirements that are typically stated as shall, will, or should requirements • Requirements of this type include capability, functional, behavioral, quality attribute, operational, safety, compliance, regulatory, logistic, training, etc. requirements that the government expects any proposed solutions to meet.
System Requirements	
Overview of Information Needed	**Specific Information Needed**
System use cases • Allow SFR participants to easily identify and discuss system use cases and understand the associations between specific system use cases and Government requirements, system requirements, quality attribute scenarios, operational use cases, system functions, and other system use cases. • Allow SFR participants to visualize typical interactions between specific actors and the system in support of operational use cases.	Contractors are expected to refine and decompose the primary system use cases that were identified during SRR and present any refinements to those use cases and the results of decomposing those use cases at SFR. See Section A.2.1 for the specifics of system use case information needed to support SFRs.
Operational use cases • Allow SFR participants to easily identify and discuss operational use cases and understand the associations between specific operational use cases and Government requirements, system requirements, quality attribute scenarios, system use cases, system functions, and other operational use cases. • Allow SFR participants to visualize typical day-in-the-life scenarios where system use cases are applied by one or more actors in various combinations to achieve operational goals/objectives.	Contractors are expected to refine and decompose the primary operational use cases that were identified during SRR and present any refinements to those use cases and the results of decomposing those use cases at SFR. See Section A.2.1 for the specifics of operational use case information needed to support SFRs. • Intended for operational use cases that describe "a day in the life" of a system.

System functions • Allow SFR participants to easily identify and discuss functions that will be provided by the system and understand the associations between specific system functions and Government requirements, system requirements, quality attribute scenarios, system use cases, operational use cases, and other system functions.	Contractors are expected to refine and decompose the primary system functions that were identified during SRR and present any refinements to those functions and the results of decomposing those functions at SFR. See Section A.3.1 for the specifics of system function information needed to support SFRs.
System requirements • Allow SFR participants to easily identify and discuss system requirements and understand the associations between specific system requirements and Government requirements, quality attribute scenarios, system use cases, operational use cases, system functions, and other system requirements.	Contractors are expected to refine and decompose the high-level system requirements that were identified during SRR and present any refinements to those requirements and the results of decomposing those requirements at SFR. See Section A.1.1 for the specifics of system requirements information needed to support SFRs. • Intended for requirements that are typically stated as shall, will, or should requirements • Requirements of this type include functional, behavioral, and quality attribute requirements.
External interfaces • Allow SFR participants to easily identify and discuss requirements for external interfaces between the system and external systems.	Contractors are expected to provide definitions for interfaces that were identified during SRR and present those definitions at SFR. Identification and description of internal system interfaces is not expected until PDR. See Section A.4.1 for the specifics of external interface information needed to support SFRs.

6 Software Specification Reviews (SSR) – <section placeholder>

This section provides an overview of the information that is needed to adequately support an SSR. Specific information that is needed to support SSRs and suggestions for what techniques, languages/frameworks, and tools might be appropriate for modeling that information is outlined in Appendix A. References to the material in Appendix A are provided in this section as appropriate.

6.1 What is an SSR?

SSRs are conducted to:

- <TBD>
- <TBD>

6.2 What Information is needed to achieve an SSR's stated purpose?

Information needed to support the stated purpose of an SSR is described in the tables below.

Table 5 – Information Needed for SSR

<TBD>	
Overview of Information Needed	**Specific Information Needed**
<TBD> • Allow SSR participants to <TBD>.	<statement of expectations> See Section <TBD> for the specifics of <TBD> information needed to support SSRs.
System Requirements (SRs)	
Overview of Information Needed	**Specific Information Needed**
<TBD> Allow SSR participants to <TBD>.	<statement of expectations> See Section <TBD> for the specifics of <TBD> information needed to support SSRs.

7 Preliminary Design Reviews (PDR) – <section placeholder>

This section provides an overview of the information that is needed to adequately support a PDR. Specific information that is needed to support PDRs and suggestions for what techniques, languages/frameworks, and tools might be appropriate for modeling that information is outlined in Appendix A. References to the material in Appendix A are provided in this section as appropriate.

7.1 What is a PDR?

PDRs are conducted to:

- <TBD>
- <TBD>

7.2 What Information is needed to achieve an PDR's stated purpose?

Information needed to support the stated purpose of a PDR is described in the tables below.

Table 6 – Information Needed for PDR

<TBD>	
Overview of Information Needed	**Specific Information Needed**
<TBD> • Allow PDR participants to <TBD>.	<statement of expectations> See Section <TBD> for the specifics of <TBD> information needed to support PDRs.
System Requirements (SRs)	
Overview of Information Needed	**Specific Information Needed**
<TBD> • Allow PDR participants to <TBD>.	<statement of expectations> See Section <TBD> for the specifics of <TBD> information needed to support PDRs.

8 Critical Design Reviews (CDR) – <section placeholder>

This section provides an overview of the information that is needed to adequately support a CDR. Specific information that is needed to support CDRs and suggestions for what techniques, languages/frameworks, and tools might be appropriate for modeling that information is outlined in Appendix A. References to the material in Appendix A are provided in this section as appropriate.

8.1 What is a CDR?

CDRs are conducted to:

- <TBD>
- <TBD>

8.2 What Information is needed to achieve a CDR's stated purpose?

Information needed to support the stated purpose of a CDR is described in the tables below.

Table 7 – Information Needed for CDR

<TBD>	
Overview of Information Needed	**Specific Information Needed**
<TBD> • Allow CDR participants to <TBD>.	<statement of expectations> See Section <TBD> for the specifics of <TBD> information needed to support CDRs.
System Requirements (SRs)	
Overview of Information Needed	**Specific Information Needed**
<TBD> • Allow CDR participants to <TBD>.	<statement of expectations> See Section <TBD> for the specifics of <TBD> information needed to support CDRs.

Appendix A Information Needed to Support Lifecycle Events

This appendix describes what information is needed to support various lifecycle events and provides suggestions for how to model that information. Each subsection focuses on a particular type of information (e.g., Section A.1 focuses on requirement information). Each subsection begins by identifying the essential and non-essential information needed to support one or more lifecycle events and ends with suggestions on how to model that information.

Modeling suggestions in each subsection are presented using the following table:

Table 8 - Template for Modeling Suggestions

Models to Consider for [Info Type] Information				Modeling Frameworks		Modeling Languages			Modeling Tools					
				DoDAF	UAF	SysML	UML	AADL	MS Office	Cameo EA	Enterprise Architect	Rhapsody	DOORS	OSATE/AADL
Modeling Categories	**E**	**C**	**Modeling Techniques**											
Behavioral			Activity diagrams			x	x			x	x	x		
			Communication diagrams				x			x	x	x		
			Dataflow diagrams								x			
			Interaction overview diagrams				x			x	x			
			Sequence diagrams			x	x			x	x	x		
			State machine diagrams			x	x	x		x	x	x		x
			Timing diagrams				x				x	x		
			Use case diagrams			x	x			x	x	x		
			Simulations									x		
Structural			Block definition diagrams			x				x	x	x		
			Class diagrams				x			x	x	x		
			Component diagrams				x	x		x	x	x		x
			Composite structure diagrams				x	x		x	x	x		x
			Deployment diagrams				x	x		x	x	x		x
			Internal block diagrams			x				x	x	x		
			Object diagrams				x	x		x	x	x		x
			Package diagrams			x	x	x		x	x	x		x
Textual			Mission threads											
			Quality attribute scenarios											
			Tables/matrices						x		x	x	x	
			Requirement databases										x	
			Descriptive paragraphs			x	x		x	x	x	x	x	
Associational			Allocation tables/matrices			x				x		x	x	
			Allocation diagrams					x				x		x
			Dependency tables/matrices									x	x	
			Requirement diagrams			x				x	x	x		

Legend	
x	Directly supported
	Indirectly supported (via use of supported artifacts)
	Create artifacts using generic shape library
	Insert or link to externally created artifacts
	Not supported
*	Suggestion for modeling technique. Applies to column E (essential) and column C (complements).

The tables identifies various techniques that are commonly used for software and system modeling and indicates to what extent those techniques are supported by various frameworks, languages, and tools that are commonly used for modeling. With the exception of the columns E (which stands for "essential") and C (which stands for "complements"), all information the table remains the same regardless of where the table is used. The content of columns E and C directly depend on where the table is used. For example, the table is used in Section A.1 to provide modeling suggestions for requirement information

and used again in Section A.4 to provide modeling suggestions for interface information. With the exception of the E and C columns, the two tables are identical.

Modeling categories are used in the table to group techniques into the types of models that will result when those techniques are applied, namely, behavioral, structural, textual, and associational models. Each technique is cross-referenced with the modeling frameworks, languages, and tools listed in the table to indicate to what extent a technique is supported by each. Column E offers suggestions for one or more techniques to use for modeling information that is noted as essential to one or more lifecycle events. Column C offers suggestions on additional models one might consider using to complement any models developed for essential information.

A.1 Requirement Information

A.1.1 Essential Requirement Information

Specific information about government requirements, system requirements, and various associations that is essential to achieving the stated purposes of SRRs and SFRs is provided in Table 9. Information in the table marked with "SRR" and/or "SFR" in the "Essential" column is considered essential for every requirement at the corresponding event(s). Information in the table marked '-' in the "Essential" column may be useful but is not considered essential for SRR or SFR (but may be marked as essential for SSR, PDR, and/or CDR in subsequent revisions of this document).

Table 9 - Essential Requirement Information

Information Needed	Essential	Description
Requirement ID	SRR, SFR	Unique identifier (e.g., requirement number).
Requirement description	SRR, SFR	Textual description of the requirement.
Requirement rationale	SRR, SFR	Provides a rationale for the requirement (e.g., the rationale for system a system constraint like "The C++ programming language shall be used for software development." might be "All of our developers only know how to program in C++. Choosing any other programming language w ill significantly increase cost and schedule").
Requirement categories	_	One or more logical requirement categories to w hich the requirement belongs. For example, the requirement "The life preserver shall inflate to design shape w ithin 10 seconds." might fall into both the performance and safety requirement categories.
Ow ner	_	The source of the requirement (e.g., person, organization).
Qualification provisions	_	One or more methods that w ill be used to ensure that the requirement has been met.
Status	_	Indicates the current state of the requirement (e.g., approved, tentative, tested)
Risks	_	The risks associated w ith the requirement.
Risk assessments	_	One or more risk assessments related to the requirement.
Other requirement attributes	_	Other program-specific/project-specific requirement attributes of interest that have not been presented in this table (e.g., Block/Increment release target).
Associations[1] to parent requirements	SRR, SFR	Links the requirement to its parent requirement(s). See Section A.5
Associations to child requirements	SRR, SFR	Links from the requirement to its child requirements. See Section A.5
Associations to other requirements	SRR, SFR	Links from the requirement to other requirements (e.g., links betw een a system requirement and one or more government requirements). See Section A.5
Associations to other artifacts	SRR, SFR	Links from the requirement to one or more related artifacts (e.g., a requirement may be linked to related standards documents and/or modeling artifacts that clarify the requirement). See Section A.5
Associations to use cases	SRR, SFR	Links from the requirement to one or more system and/or operational use cases. See Section A.5

[1] Although at liberty to name the various associations listed in this and other tables that appear in this appendix to best suit their needs, contractors must provide a key that shows all association names and descriptions for what the associations mean, allow, and/or show.

Information Needed	Essential	Description
Associations to system functions	SRR, SFR	Links from the requirement to one or more related system functions. See Section A.5
Associations to quality attribute scenarios	SRR, SFR	Links from the requirement to one or more related quality attribute scenarios. See Section A.5

A.1.2 Modeling Requirement Information

Table 10 provides suggestions in column E for modeling requirement information noted as essential in in Section A.1.1. The table also provides suggestions in column C on additional models to consider adding to complement any models developed for essential information.

Table 10 – Models to Consider for Requirement Information

Models to Consider for Requirement Information				Modeling Frameworks		Modeling Languages			Modeling Tools					
				DoDAF	UAF	SysML	UML	AADL	MS Office	Cameo EA	Enterprise Architect	Rhapsody	DOORS	OSATE/AADL
Modeling Categories	**E**	**C**	**Modeling Techniques**											
Behavioral		*	Activity diagrams			x	x			x	x	x		
		*	Communication diagrams				x			x	x	x		
		*	Dataflow diagrams								x			
		*	Interaction overview diagrams				x			x	x			
		*	Sequence diagrams			x	x			x	x	x		
		*	State machine diagrams			x	x	x		x	x	x		x
		*	Timing diagrams				x				x	x		
		*	Use case diagrams			x	x			x	x	x		
		*	Simulations									x		
Structural			Block definition diagrams			x				x	x	x		
		*	Class diagrams				x			x	x	x		
			Component diagrams				x	x		x	x	x		x
			Composite structure diagrams				x	x		x	x	x		x
			Deployment diagrams				x	x		x	x	x		x
			Internal block diagrams			x				x	x	x		
		*	Object diagrams				x	x		x	x	x		x
		*	Package diagrams			x	x	x		x	x	x		x
Textual		*	Mission threads											
		*	Quality attribute scenarios											
	*		Tables/matrices						x		x	x	x	
	*		Requirement databases										x	
	*		Descriptive paragraphs			x	x		x	x	x	x	x	
Associational	*		Allocation tables/matrices			x				x		x	x	
	*		Allocation diagrams					x				x		x
	*		Dependency tables/matrices									x	x	
	*		Requirement diagrams			x				x	x	x		

Legend	
x	Directly supported
	Indirectly supported (via use of supported artifacts)
	Create artifacts using generic shape library
	Insert or link to externally created artifacts
	Not supported
*	Suggestion for modeling technique. Applies to column E (essential) and column C (complements).

Requirements stated primarily as text in the "shall/will/should" structure are typically difficult to describe a sense of behavioral characteristic that may be crucial to the system being developed. Also, structural characteristics for the system being developed may be impactful for certain requirements in ways not easily represented in textual form alone. Judicious use of supporting diagrams from the Behavioral and Structural modeling categories can be very effective at clarifying the intended requirement(s).

A.2 Use Case Information

A.2.1 Essential Use Case Information

Specific information system use cases, operational use cases, and various associations that is essential to achieving the stated purposes of SRRs and SFRs is provided in Table 11. Information in the table marked with "SRR" and/or "SFR" in the "Essential" column is considered essential for every use case at the corresponding event(s). Information in the table marked '-' in the "Essential" column may be useful but is not considered essential for SRR or SFR (but may be marked as essential for SSR, PDR, and/or CDR in subsequent revisions of this document).

Table 11 - Essential Use Case Information

Information Needed	Essential	Description
Use case ID	SRR, SFR	Unique identifier (e.g., use case number).
Use case description	SRR, SFR	Textual description of the use case.
Use case rationale	_	Provides a rationale for the use case
Requirement categories	_	One or more logical requirement categories to which the use case belongs.
Owner	_	The source of the use case (e.g., person, organization).
Qualification provisions	_	One or more methods that will be used to ensure that the use case has been met.
Status	_	Indicates the current state of the use case (e.g., approved, tentative, tested)
Risks	_	The risks associated with the use case.
Risk assessments	_	One or more risk assessments related to the use case.
Other use case attributes	_	Other program-specific/project-specific use case attributes of interest that have not been presented in this table (e.g., Block/Increment release target).
Associations to other use cases	SRR, SFR	Links from the use case to other use cases (e.g., a use case is linked to one or more use cases that are included in or extent the use case). See Section A.5
Associations to other requirements	SRR, SFR	Links from the use case to other requirements (e.g., links between a use case and one or more system requirements and/or operational requirements). See Section A.5
Associations to design elements	_	Links from the use case to one or more design elements (e.g., a link between a use case and the flight control subsystem indicating "This use case has been allocated to the flight control subsystem." or a link between a use case and a particular software module). See Section A.5
Associations to other artifacts	SRR, SFR	Links from the use case to one or more related artifacts (e.g., a use case may be linked to related standards documents and/or modeling artifacts that clarify the use case). See Section A.5

A.2.2 Modeling Use Case Information

Table 12 provides suggestions in column E for modeling use case information noted as essential in in Section A.2.1. The table also provides suggestions in column C on additional models to consider adding to complement any models developed for essential information.

Table 12 – Models to Consider for Use Case Information

Models to Consider for Use Case Information				Modeling Frameworks		Modeling Languages			Modeling Tools					
				DoDAF	UAF	SysML	UML	AADL	MS Office	Cameo EA	Enterprise Architect	Rhapsody	DOORS	OSATE/AADL
Modeling Categories	**E**	**C**	**Modeling Techniques**											
Behavioral		*	Activity diagrams			x	x			x	x	x		
		*	Communication diagrams				x			x	x	x		
		*	Dataflow diagrams								x			
		*	Interaction overview diagrams				x			x	x			
		*	Sequence diagrams			x	x			x	x	x		
		*	State machine diagrams			x	x	x		x	x	x		x
		*	Timing diagrams				x				x	x		
	*		Use case diagrams			x	x			x	x	x		
		*	Simulations									x		
Structural			Block definition diagrams			x				x	x	x		
		*	Class diagrams				x			x	x	x		
			Component diagrams				x	x		x	x	x		x
			Composite structure diagrams				x	x		x	x	x		x
			Deployment diagrams				x	x		x	x	x		x
			Internal block diagrams			x				x	x	x		
		*	Object diagrams				x	x		x	x	x		x
		*	Package diagrams			x	x	x		x	x	x		x
Textual		*	Mission threads											
		*	Quality attribute scenarios											
	*		Tables/matrices						x		x	x	x	
			Requirement databases										x	
	*		Descriptive paragraphs			x	x		x	x	x	x	x	
Associational		*	Allocation tables/matrices			x				x		x	x	
		*	Allocation diagrams					x				x		x
		*	Dependency tables/matrices									x	x	
		*	Requirement diagrams			x				x	x	x		

Legend	
x	Directly supported
	Indirectly supported (via use of supported artifacts)
	Create artifacts using generic shape library
	Insert or link to externally created artifacts
	Not supported
*	Suggestion for modeling technique. Applies to column E (essential) and column C (complements).

To extend the essential use case information, almost any modeling technique within the Behavioral Modeling category can help clarify the intended behavior of the use case.

For example,

- Activity, Communications and Sequence Diagrams depict elements of work flows among actors.
- Timing and state machine diagrams overlay the temporal aspects over the sequence of work flows

A.3 System Function Information

A.3.1 Essential System Function Information

Specific information about system functions and various associations that is essential to achieving the stated purposes of SRRs and SFRs is provided in Table 13. Information in the table marked with "SRR" and/or "SFR" in the "Essential" column is considered essential for every function at the corresponding event(s). Information in the table marked '-' in the "Essential" column may be useful but is not considered essential for SRR or SFR (but may be marked as essential for SSR, PDR, and/or CDR in subsequent revisions of this document).

Table 13 - Essential System Function Information

Information Needed	Essential	Description
System Function ID	SRR, SFR	Unique identifier (e.g., system function reference number).
System Function description	SRR, SFR	Textual description of the system function. At SFR, only an initial set of high level system functions is expected to be identified
System Function category	-	One or more logical function categories to which the system function belongs. Eight generic function categories that most systems must complete over their life cycle are development, manufacturing, verification, deployment, training, operations, support, and disposal.
Qualification provisions	-	One or more methods that will be used to ensure that the system function operates as intended
Other System Function attributes	_	Other program-specific/project-specific System Function attributes of interest that have not been presented in this table (e.g., Block/Increment release target).
Associations to other System Functions	-	Links from the System Function to other System Functions (i.e., the decomposition of System Functions that establishes System Function hierarchy and establishes parent-child associations). See Section A.5
Associations to requirements	-	Links from the System Function to requirements (e.g., links between a System Function and one or more operational requirements and / or system requirements). See Section A.5
Associations to design elements	-	Links from the System Function to one or more design elements (e.g., a link between a System Function and the flight control subsystem indicating "This System Function has been allocated to the flight control subsystem." or a link between a System Function and a particular software module). See Section A.5
Associations to other artifacts	-	Links from the System Function to one or more related artifacts (e.g., a System Function may be linked to related standards documents and/or modeling artifacts that clarify the System Function). See Section A.5

A.3.2 Modeling System Function Information

Table 14 provides suggestions in column E for modeling system function information noted as essential in in Section A.3.1. The table also provides suggestions in column C on additional models to consider adding to complement any models developed for essential information.

Table 14- Models to Consider for System Function Information

Models to Consider for System Function Information				Modeling Frameworks		Modeling Languages			Modeling Tools					
				DoDAF	UAF	SysML	UML	AADL	MS Office	Cameo EA	Enterprise Architect	Rhapsody	DOORS	OSATE/AADL
Modeling Categories	**E**	**C**	**Modeling Techniques**											
Behavioral		*	Activity diagrams			x	x			x	x	x		
		*	Communication diagrams				x			x	x	x		
		*	Dataflow diagrams								x			
		*	Interaction overview diagrams				x			x	x			
		*	Sequence diagrams			x	x			x	x	x		
		*	State machine diagrams			x	x	x		x	x	x		x
		*	Timing diagrams				x				x	x		
		*	Use case diagrams			x	x			x	x	x		
		*	Simulations									x		
Structural			Block definition diagrams			x				x	x	x		
		*	Class diagrams				x			x	x	x		
			Component diagrams				x	x		x	x	x		x
			Composite structure diagrams				x	x		x	x	x		x
			Deployment diagrams				x	x		x	x	x		x
			Internal block diagrams			x				x	x	x		
		*	Object diagrams				x	x		x	x	x		x
		*	Package diagrams			x	x	x		x	x	x		x
Textual		*	Mission threads											
		*	Quality attribute scenarios											
	*		Tables/matrices						x		x	x	x	
	*		Requirement databases										x	
	*		Descriptive paragraphs			x	x		x	x	x	x	x	
Associational		*	Allocation tables/matrices			x				x		x	x	
		*	Allocation diagrams					x				x		x
		*	Dependency tables/matrices									x	x	
		*	Requirement diagrams			x				x	x	x		

Legend	
x	Directly supported
	Indirectly supported (via use of supported artifacts)
	Create artifacts using generic shape library
	Insert or link to externally created artifacts
	Not supported
*	Suggestion for modeling technique. Applies to column E (essential) and column C (complements).

System functions are behaviors carried out by a system in response to various inputs. Complementing system function descriptions with behavioral, structural, and associational models helps system stakeholders to better understand those functions and to understand the relationship of system functions to requirements.

A.4 Interface Information

A.4.1 Essential Interface Information

Specific information about system interfaces and various associations that is essential to achieving the stated purposes of SRRs and SFRs is provided in Table 15. Information in the table marked with "SRR" and/or "SFR" in the "Essential" column is considered essential for every interface at the corresponding event(s). Information in the table marked '-' in the "Essential" column may be useful but is not considered essential for SRR or SFR (but may be marked as essential for SSR, PDR, and/or CDR in subsequent revisions of this document).

Table 15 - Essential Interface Information

Information Needed	Essential	Description
Interface ID	SRR, SFR	Unique identifier (e.g., Interface number).
Interface name	SRR, SFR	Textual name given to the element (e.g., Interface name)
Interface description	SRR, SFR	Textual description of the interface.
Proprietary	SRR, SFR	Indicates whether the interface is proprietary or non-proprietary.
Associations to requirements		Links from the interface description to one or more Government or System requirements. See Section A.5
Associations to other artifacts		Links from the interface description to one or more related artifacts (e.g., an interface description may be linked to related standards documents) See Section A.5

A.4.2 Modeling Interface Information

Table 16 provides suggestions in column E for modeling interface information noted as essential in in Section A.4.2. The table also provides suggestions in column C on additional models to consider adding to complement any models developed for essential information.

Table 16 – Models to Consider for Interface Information

Models to Consider for Interface Information				Modeling Frameworks		Modeling Languages			Modeling Tools					
				DoDAF	UAF	SysML	UML	AADL	MS Office	Cameo EA	Enterprise Architect	Rhapsody	DOORS	OSATE/AADL
Modeling Categories	**E**	**C**	**Modeling Techniques**											
Behavioral		*	Activity diagrams			x	x			x	x	x		
		*	Communication diagrams				x			x	x	x		
		*	Dataflow diagrams								x			
		*	Interaction overview diagrams				x			x	x			
		*	Sequence diagrams			x	x			x	x	x		
		*	State machine diagrams			x	x	x		x	x	x		x
		*	Timing diagrams				x				x	x		
		*	Use case diagrams			x	x			x	x	x		
		*	Simulations									x		
Structural			Block definition diagrams			x				x	x	x		
		*	Class diagrams				x			x	x	x		
			Component diagrams				x	x		x	x	x		x
			Composite structure diagrams				x	x		x	x	x		x
			Deployment diagrams				x	x		x	x	x		x
			Internal block diagrams			x				x	x	x		
		*	Object diagrams				x	x		x	x	x		x
		*	Package diagrams			x	x	x		x	x	x		x
Textual		*	Mission threads											
		*	Quality attribute scenarios											
	*		Tables/matrices						x		x	x	x	
	*		Requirement databases										x	
	*		Descriptive paragraphs			x	x		x	x	x	x	x	
Associational		*	Allocation tables/matrices			x				x		x	x	
		*	Allocation diagrams					x				x		x
		*	Dependency tables/matrices									x	x	
		*	Requirement diagrams			x				x	x	x		

Legend	
x	Directly supported
	Indirectly supported (via use of supported artifacts)
	Create artifacts using generic shape library
	Insert or link to externally created artifacts
	Not supported
*	Suggestion for modeling technique. Applies to column E (essential) and column C (complements).

Interface requirements represent the physical and logical connections among various parts of the system being developed as well as connections among various external actors and the system being developed. An interface may exist between two entities either within the same system or between two separate systems. An interface may be implemented to a specific standard, it may be a physical-only interface, or it may be a software interface with complex protocols and data exchanges. An interface's topology may be point-to-point, point to multi-point, a bus structure, a publish-subscribe structure, etc. Further, interfaces may be open or proprietary.

Due to the broad range of attributes that describe a specific interface, utilizing Modeling Techniques from different Modeling Categories from the table immediately above to complement the interface requirements are imperative to clearly describe the

operation of an interface and to achieve interoperability among the systems and system elements. The Modeling Technique(s) chosen are highly program and interface dependent. Often, multiple Modeling techniques across several Modeling Categories are needed to clearly describe the interface. Introducing these Techniques early in a program lifecycle can significantly increase clarity and understanding of interfaces.

A.5 Association Information

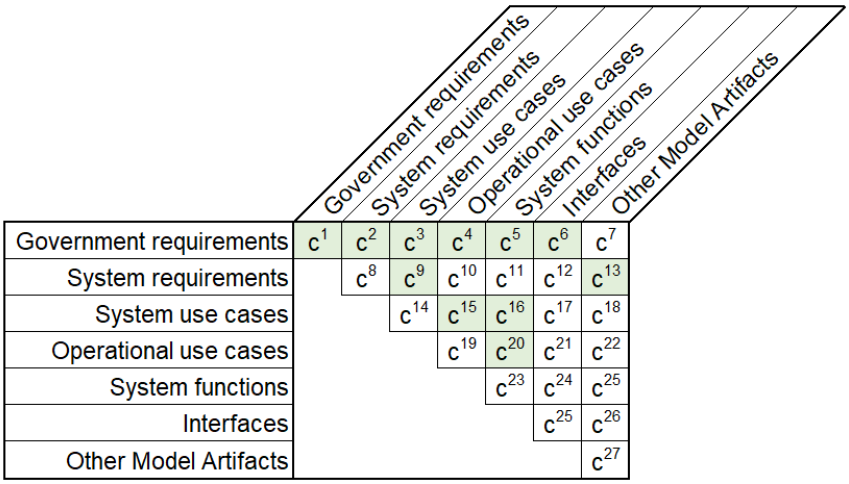
This section describes the types of associations that are considered essential to supporting SRRs and SFRs.

Before delving to "essential associations", it is important to consider associations in general. Associations can be made between any two types of model elements. For example, a model can have associations between two requirements, two software elements, a requirement and a software element, a software element and a hardware element, two model diagrams, two model attributes, and even an association between two associations! Furthermore, given two instances of any types of model element, many different types of associations between the two instances are possible. For example[2], possible associations between two requirement instances include the "parent/child", "derived from", and "refines" associations; Possible associations between a requirement instance and a software element instance include the "allocated to", "implements", and "satisfies" associations. Certainly not all possible types of associations between all possible types of model elements will prove useful. Whether or not any particular type of association makes sense and proves useful depends on a number of key factors including (but not limited to): 1) the name given to the association, 2) the meaning of the association, 3) the context in which the association is used, 4) who (if anyone) needs to know about the association, and 5) being able to easily determine which model elements are associated with each other via the association.

The above being said, we will limit the scope of this discussion to types of associations that are needed to support the lifecycle events described in this document thus far[3], namely, SRR and SFR.

Table 17 shows the types of associations[4] that are considered essential to supporting SRRs (highlighted in green).

Table 17 – Associations Needed for SRRs



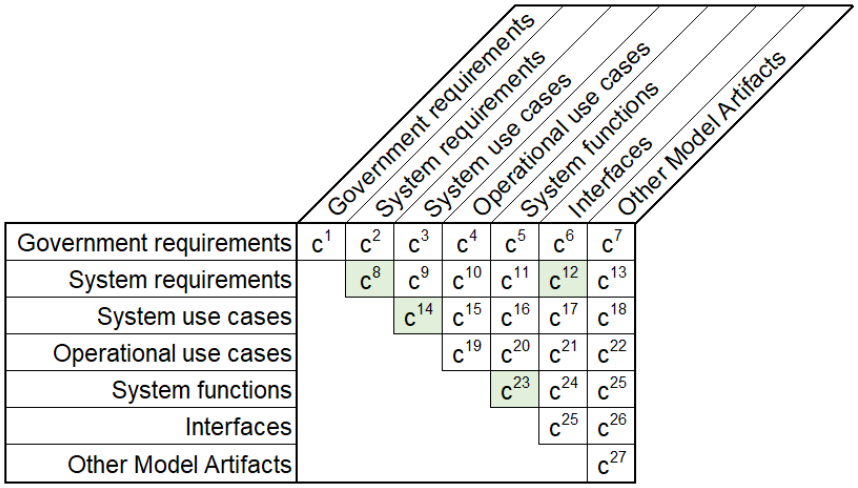
Association	Association Description	Event Needed
C1	Indicates a parent/child relationship between one or more Government requirements.	SRR
C2	Indicates that a high-level system requirement was introduced in response to one or more Government requirements.	SRR
C3	Indicates that a high-level system use case was introduced in response to one or more Government requirements.	SRR

[2] Examples of associations names listed here are not intended as examples of associations defined by any particular modeling language. Any name can be given to an association regardless of whether or not a modeling language exists that defines that particular association name.
[3] We hope that this work will continue and the document will eventually include modeling suggestions for SSR, PDR, and CDR.
[4] Association names should always indicate the types of relationships they represent. Although at liberty to name the associations to best suit their needs, contractors must provide a key that shows all association names and descriptions for what the associations mean, allow, and/or show.

Association	Association Description	Event Needed
C4	Indicates that a high-level operational use case was introduced in response to one or more Government requirements.	SRR
C5	Indicates that a high-level system function was introduced in response to one or more Government requirements.	SRR
C6	Indicates that an interface was introduced in response to one or more Government requirements.	SRR
C9	Indicates that a high-level system use case is refined by one or more high-level system requirements.	SRR
C13	Indicates that a high-level system requirement is refined by one or more model artifacts (e.g., "The system shall provide an operator training mode." is refined by a state diagram which shows the relationship between operational mode and training mode and the activities that can be performed by an operator in training mode).	SRR
C15	Indicates that a high-level operational use case comprises one or more system use cases.	SRR
C16	Indicates that a high-level system use case is supported by one or more high-level system functions.	SRR
C20	Indicates that a high-level operational use case is supported by one or more high-level system functions.	SRR

Table 18 shows the types of associations that are considered essential to supporting SFRs (highlighted in green).

Table 18 – Associations Needed for SFRs



Association	Association Description	Event Needed
C8	Indicates a parent/child relationship between one or more system requirements. Contractors are expected to refine and decompose the high-level system requirements that were identified during SRR and present any refinements to those requirements and the results of decomposing those requirements at SFR.	SFR
C12	Indicates that an interface requirement identified during SRR is associated with an interface definition. Contractors are expected to refine interface definitions that were identified during SRR and present any refinements to those definitions at SFR. Identification and description of internal system interfaces is not expected until PDR.	SFR
C14	Indicates use cases that are refinements to high-level system use case that were identified at SRR.	SFR

Association	Association Description	Event Needed
C23	Indicates a parent/child relationship between one or more system functions. Contractors are expected to refine and decompose the primary system functions that were identified during SRR and present any refinements to those functions and the results of decomposing those functions at SFR.	SFR

Appendix B Modeling Framework, Language, and Tool Overviews

This appendix provides overviews for the modeling frameworks, languages, and tools referred to in this document.

B.1 Modeling Frameworks

B.1.1 Department of Defense Architecture Framework (DoDAF)

(See https://acqnotes.com/acqnote/tasks/architecting-overview)

In the Department of Defense (DoD), the development of an architecture for a system is called the DoD Architecture Framework (DoDAF). DoDAF is the overarching, comprehensive framework and conceptual model enabling the development of architectures for DoD systems. The DoDAF serves as one of the principal pillars supporting the DoD Chief Information Officer (CIO) in his responsibilities for the development and maintenance of architectures required under the Clinger-Cohen Act.

DoDAF Steps

There are 6 steps that make up the DoDAF Design Process. These steps are:

- Step 1: Determine Intended Use of Architecture
- Step 2: Determine Scope of Architecture
- Step 3: Determine Data Required to Support Architecture Development
- Step 4: Collect, Organize, Correlate, and Store Architectural Data
- Step 5: Conduct Analyses in Support of Architecture Objectives
- Step 6: Document Results in Accordance with Decision-Maker Needs

DoDAF Main References:

- Guide: DoDAF Architecture Framework Version 2.02
- Website: https://dodcio.defense.gov/Library/DoD-Architecture-Framework/

DoDAF Viewpoints

The documented results of the DoDAF process are organized into the following Viewpoints (aka Architecture Types):

- All Viewpoint (AV): describes the overarching aspects of architecture context that relate to all viewpoints.
- Capability Viewpoint (CV): articulates the capability requirements, the delivery timing, and the deployed capability.
- Data and Information Viewpoint (DIV): articulate the data relationships and alignment structures in the architecture content for the capability and operational requirements, system engineering processes, and systems and services.
- Operational Viewpoint (OV): includes the operational scenarios, activities, and requirements that support capabilities.
- Project Viewpoint (PV): describes the relationships between operational and capability requirements and the various projects being implemented.
- Services Viewpoint (SvcV): is the design for solutions articulating the Performers, Activities, Services, and their Exchanges, providing for or supporting operational and capability functions.

- Standards Viewpoint (StdV): articulate the applicable operational, business, technical, and industry policies, standards, guidance, constraints, and forecasts that apply to capability and operational requirements, system engineering processes, and systems and services.
- Systems Viewpoint (SV): for Legacy support, is the design for solutions articulating the systems, their composition, interconnectivity, and context providing for or supporting operational and capability functions.

B.1.2 Unified Architecture Framework (UAF)

(See https://www.omgwiki.org/uaf/doku.php)

Background

UAF evolved from the Unified Profile for Department of Defense Architecture Framework (DoDAF) and Ministry of Defense Architecture Framework (MODAF) (UPDM), version 2.1. UPDM is not a new architectural framework- it merges pre-existing concepts from DoDAF and MODAF into a combined metamodel. UPDM has been adopted by a wide variety of organizations, extending beyond the military and into federal and industry applications. Because of this, there was an increasing demand for UPDM to become more industrialized and to support other frameworks other than just DoDAF and MODAF. This led to the creation of UAF.

What is UAF?

The Unified Architecture Framework (UAF) is an architecture framework that provides visualization for specific stakeholders concerns through engineering domains organized by various views. The views are artifacts for visualizing, understanding, and assimilating the broad scope and complexities of an architecture description through tabular or graphical means. UAFP enables the extraction of specified and custom views from an integrated architecture description (AD) in support of a model-based systems engineering (MBSE) approach. The views describe a system from a set of stakeholders' concerns such as security or information. The UAFP specification supports the Department of Defense Architecture Framework (DoDAF) 2.02, the Ministry of Defense Architecture Framework (MODAF), Security Views from Canada's Department of National Defense Architecture Framework (DNDAF) and the North Atlantic Treaty Organization (NATO) Architecture Framework (NAF) v 4. The core concepts in the UAF domain metamodel used to specify the UAFP are based upon the DoDAF 2.0.2 Domain Metamodel (DM2) and the MODAF ontological data exchange mechanism (MODEM, which is intended to provide the basis for the next version of NAF). The intent is to provide a standard representation for AD support for Industry, Government, and Defense Organizations. ADs as part of their Systems Engineering (SE) technical processes. UAFP supports the capability to:

- model architectures for a broad range of complex systems, which may include hardware, software, data, personnel, and facility elements;
- model consistent architectures for system-of-systems (SoS) down to lower levels of design and implementation;
- support the analysis, specification, design, and verification of complex systems;
- support cybersecurity analysis, specification, and mitigation of security risks from a system/infrastructure perspective and to aggregate the impact analysis to the operational perspective and cybersecurity risks' impact on the mission; and
- improve the ability to exchange architecture information among related tools that are SysML based and tools that are based on other standards.

B.2 Modeling Languages

B.2.1 Architecture Analysis and Design Language (AADL)

(See https://en.wikipedia.org/wiki/Architecture_Analysis_%26_Design_Language)

Architecture Analysis and Design Language (AADL) is a formal notation for the modeling and analysis of real-time safety-critical embedded systems where sensors and actuators are tightly coupled with software components. It facilitates the analysis of interactions between hardware and software components. It focuses on system design specification using a rich, formal semantics that can be used to generate and analyze the system. It is a SAE International Aerospace Standard Suite (AS-5506 series). It is both a textual and graphical language that uses component-based modelling concepts that has precise semantics.

An AADL model contains component types and implementation with their interfaces, subcomponents, and other properties. It defines a system in a hierarchical manner, with a top component called the root system and other component categories are grouped into three clusters: hardware, software, and physical/hybrid. Hardware components include processors, memory, buses, and virtual versions of each. Software components include process, thread, subprograms, and data. Physical system components are modeled as devices. System components can contain collection of hardware, software, and physical components that when connected can form an entire system.

The AADL contains several property types that can be extended to support user domain specific properties. The component structure along with their property annotations are instantiated and is the basis for performing many system analysis in the areas of performance, safety, and security.

The development and analysis of AADL models is supported by the Open Source AADL Tool Environment (OSATE) developed and maintained by the SEI. OSATE contains a syntax-aware text editor and synchronized graphical editor. AADL models are organized in separate projects in a workspace. The tool supports validation of AADL models according to all naming and legality rules defined in the AADL standard. OSATE contains approximately 15 analysis tools and supports plugins from third parties as well as user defined plugins.

The AADL standard:

- gives you the power to specify and generate a single model that can be analyzed for multiple qualities
- provides an industry-standard, textual and graphic notation with precise semantics to model applications and execution platforms
- features an XML interchange format that supports the exchange of models between subcontractors, integrators, and agencies
- includes a UML profile that presents AADL as a specialized modeling notation within UML framework
- is supported by commercial and open source tool solutions

Benefits

The SAE AADL standard can lower development and maintenance costs by

- providing a standard, precise syntax and semantics for performance-critical systems, so that documentation can be well defined
- providing the ability to model large-scale (multi-contractor) architectures from many aspects in a single analyzable model that can be incrementally refined
- capturing the "architectural API" needed to evaluate the effect of change, such as the emergent properties of integration (e.g., safety, schedulability, end-to-end latency, and security)

- allowing early and life-cycle tracking of modeling and analysis
- analyzing the system structure and runtime behavior rather than functional behavior, complementing functional simulation
- providing a great complement to reference architectures and component-based or product-line development

B.2.2 Unified Modeling Language (UML)

(See Unified Modeling Language - Wikipedia)

The Unified Modeling Language (UML) is a general-purpose, developmental, modeling language in the field of software engineering that is intended to provide a standard way to visualize the design of a system.

The creation of UML was originally motivated by the desire to standardize the disparate notational systems and approaches to software design. It was developed at Rational Software in 1994–1995, with further development led by them through 1996.

In 1997, UML was adopted as a standard by the Object Management Group (OMG), and has been managed by this organization ever since. In 2005, UML was also published by the International Organization for Standardization (ISO) as an approved ISO standard. Since then the standard has been periodically revised to cover the latest revision of UML. In software engineering, most practitioners do not use UML, but instead produce informal hand drawn diagrams; these diagrams, however, often include elements from UML.

UML Modeling

It is important to distinguish between the UML model and the set of diagrams of a system. A diagram is a partial graphic representation of a system's model. The set of diagrams need not completely cover the model and deleting a diagram does not change the model. The model may also contain documentation that drives the model elements and diagrams (such as written use cases).

UML diagrams represent two different views of a system model:

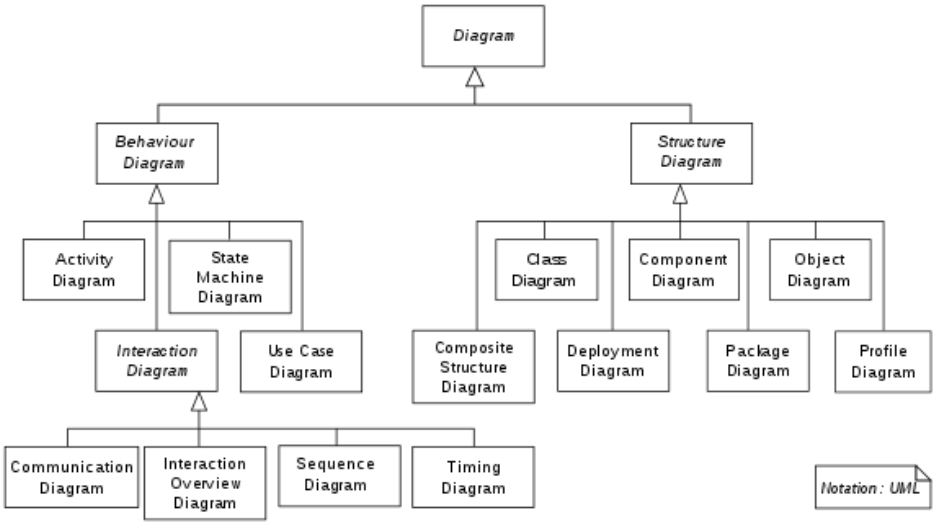
- Static (or structural) view: emphasizes the static structure of the system using objects, attributes, operations and relationships. It includes class diagrams and composite structure diagrams.
- Dynamic (or behavioral) view: emphasizes the dynamic behavior of the system by showing collaborations among objects and changes to the internal states of objects. This view includes sequence diagrams, activity diagrams and state machine diagrams.

UML models can be exchanged among UML tools by using the XML Metadata Interchange (XMI) format.

In UML, one of the key tools for behavior modeling is the use-case model, caused by Object Oriented Software Engineering (OOSE). Use cases are a way of specifying required usages of a system. Typically, they are used to capture the requirements of a system, that is, what a system is supposed to do.

UML Diagrams

UML 2 has many types of diagrams, which are divided into two categories. Some types represent *structural* information, and the rest represent general types of *behavior*, including a few that represent different aspects of *interactions*. These diagrams can be categorized hierarchically as shown in the following class diagram. Diagrams may all contain comments or notes explaining usage, constraint, or intent.



B.2.3 Systems Modeling Language (SysML)

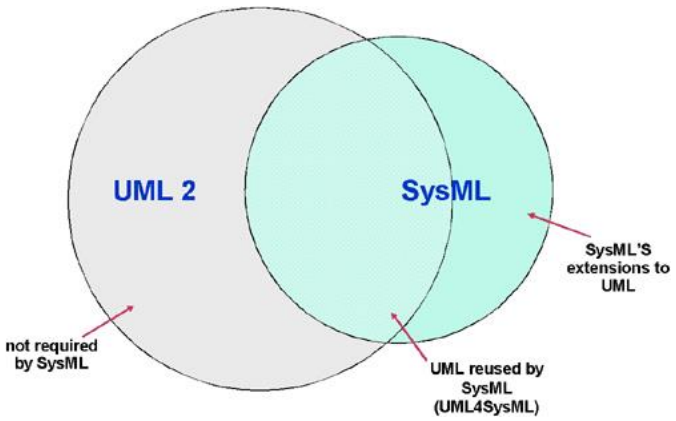
(See https://www.omgsysml.org/what-is-sysml.htm)

The OMG Systems Modeling Language™ (OMG SysML®) is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities. In particular, the language provides graphical representations with a semantic foundation for modeling system requirements, behavior, structure, and parametrics, which is used to integrate with other engineering analysis models. It represents a subset of UML 2 with extensions needed to satisfy the requirements of the UML™ for Systems Engineering RFP as indicated in Figure 1. SysML leverages the OMG XML Metadata Interchange (XMI®) to exchange modeling data between tools, and is also intended to be compatible with the evolving ISO 10303-233 systems engineering data interchange standard.

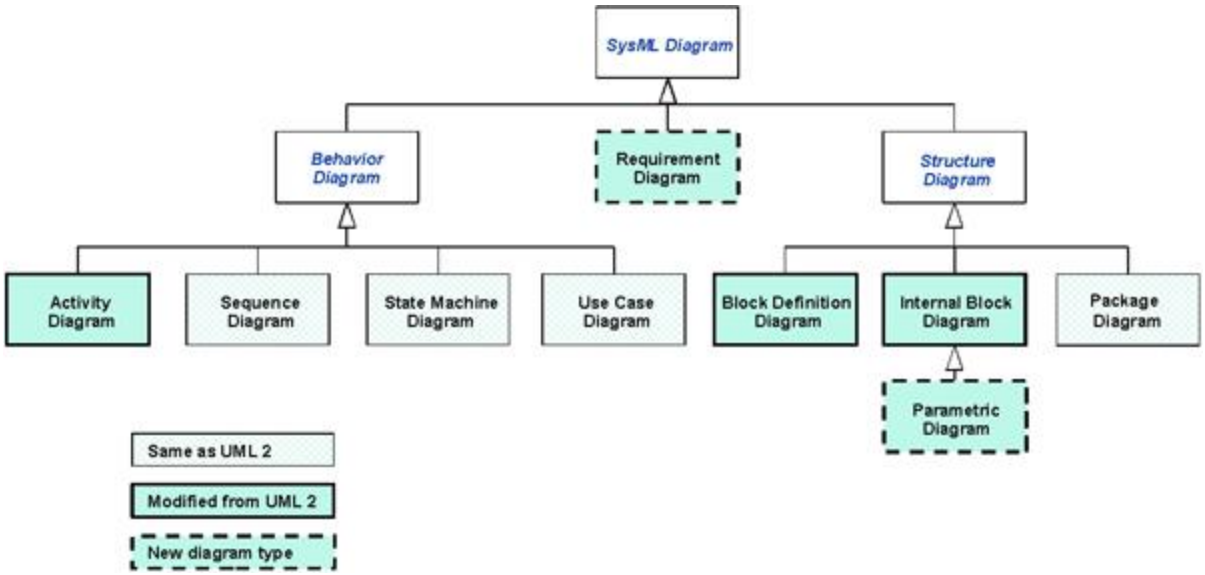
The UML for Systems Engineering RFP was developed jointly by the OMG and the International Council on Systems Engineering (INCOSE) and issued by the OMG in March 2003. The RFP specified the requirements for extending UML to support the needs of the systems engineering community. The SysML Specification was developed in response to these requirements by the diverse group of tool vendors, end users, academia, and government representatives. The Object Management Group announced the adoption on July 6, 2006 and the availability of OMG SysML™ v1.0 in September 2007.

The following diagram shows the relationship between UML and SysML:

SysML Diagram Summary



The SysML diagram types are summarized below. Refer to the OMG SysML Tutorial for an overview of the language or the APL MBSE with SysML course material for a more detailed description.



B.3 Modeling Tools

B.3.1 Enterprise Architect

(See https://en.wikipedia.org/wiki/Enterprise_Architect_(software))

Sparx Systems Enterprise Architect is a visual modeling and design tool based on the OMG UML. The platform supports: the design and construction of software systems; modeling business processes; and modeling industry based domains. It is used by businesses and organizations to not only model the architecture of their systems, but to process the implementation of these models across the full application development life-cycle.

Systems modeling using UML provides a basis for modeling all aspects of organizational architecture, along with the ability to provide a foundation for designing and implementing new systems or changing existing systems. The aspects that can be covered by this type of modeling range from laying out organizational or systems architectures, business process reengineering, business analysis, and service-oriented architectures and web modeling, through to application and database design and re-engineering, and development of embedded systems. Along with system modeling, Enterprise Architect covers the core aspects of the application development life-cycle, from requirements management through to design, construction, testing and maintenance phases, with support for traceability, project management and change control of these processes, as well as, facilities for model driven development of application code using an internal integrated-development platform.

The user base ranges from programmers and business analysts through to enterprise architects, in organizations ranging from small developer companies, multi-national corporations and government organizations through to international industry standards bodies. Sparx Systems initially released Enterprise Architect in 2000. Originally designed as a UML modeling tool for modeling UML 1.1, the product has evolved to include other OMG UML specifications 1.3, 2.0, 2.1, 2.3, 2.4.1 and 2.5.

B.3.2 Microsoft Office (MS Office)

(See https://www.techopedia.com/definition/20737/microsoft-office)

Microsoft Office is a suite of desktop productivity applications that is designed specifically by Microsoft for business use. It is a proprietary product of Microsoft Corporation and was first released in 1990. For decades, MS Office has been a dominant model in delivering modern office-related document-handling software environments.

Microsoft Office is available in 35 different languages and is supported by Windows, Mac and most Linux variants.

The core components of Microsoft Office are the six items present in the original package, notwithstanding the later addition of services like OneDrive and SharePoint and a web design tool called FrontPage.

The six core programs in Microsoft Office are:

- Word.
- Excel.
- PowerPoint.
- Access.
- Publisher.
- OneNote.

These could be separated into what you might call the "three greater applications" and the "three lesser applications" that receive much lower use by the average end-user.

The Word, Excel and PowerPoint applications in Microsoft Office are familiar household names, even to people who are not familiar with the details of the Office suite's evolution. They are often used by a diverse user base, for example, college

students, interns, or front line workers in IT. By contrast, someone may use Word, Excel and PowerPoint frequently, and rarely or never use Access, Publisher or OneNote.

The three major Microsoft Office pieces include the word processor (Word), the spreadsheet (Excel) and the visual presentation tool (PowerPoint.)

Access is a database management tool, while Publisher allows for the presentation of various marketing materials.

B.3.3 Open Source AADL Tool Environment (OSATE)

(See https://resources.sei.cmu.edu/asset_files/FactSheet/2017_010_001_506838.pdf)

A Tool Kit to Support Model-Based Engineering

The Open Source AADL Tool Environment (OSATE) Version 2

OSATE 2 is an Eclipse-based modeling framework for using AADL. In this environment, software architects can design and analyze models and then generate parts of the implementation code.

OSATE supports the textual and graphical representation of AADL. The textual editor features auto-indentation, auto-completion, and syntax highlighting. The graphical editor allows designers to revise the model, and it synchronizes the graphical and textual representations.

OSATE integrates several validation and analysis plug-ins. The SEI has developed tools for analyzing

- systemsafety
- systemsecurity
- performance
- flow latency
- scheduling
- resource budgeting (processor, weight, electrical power)

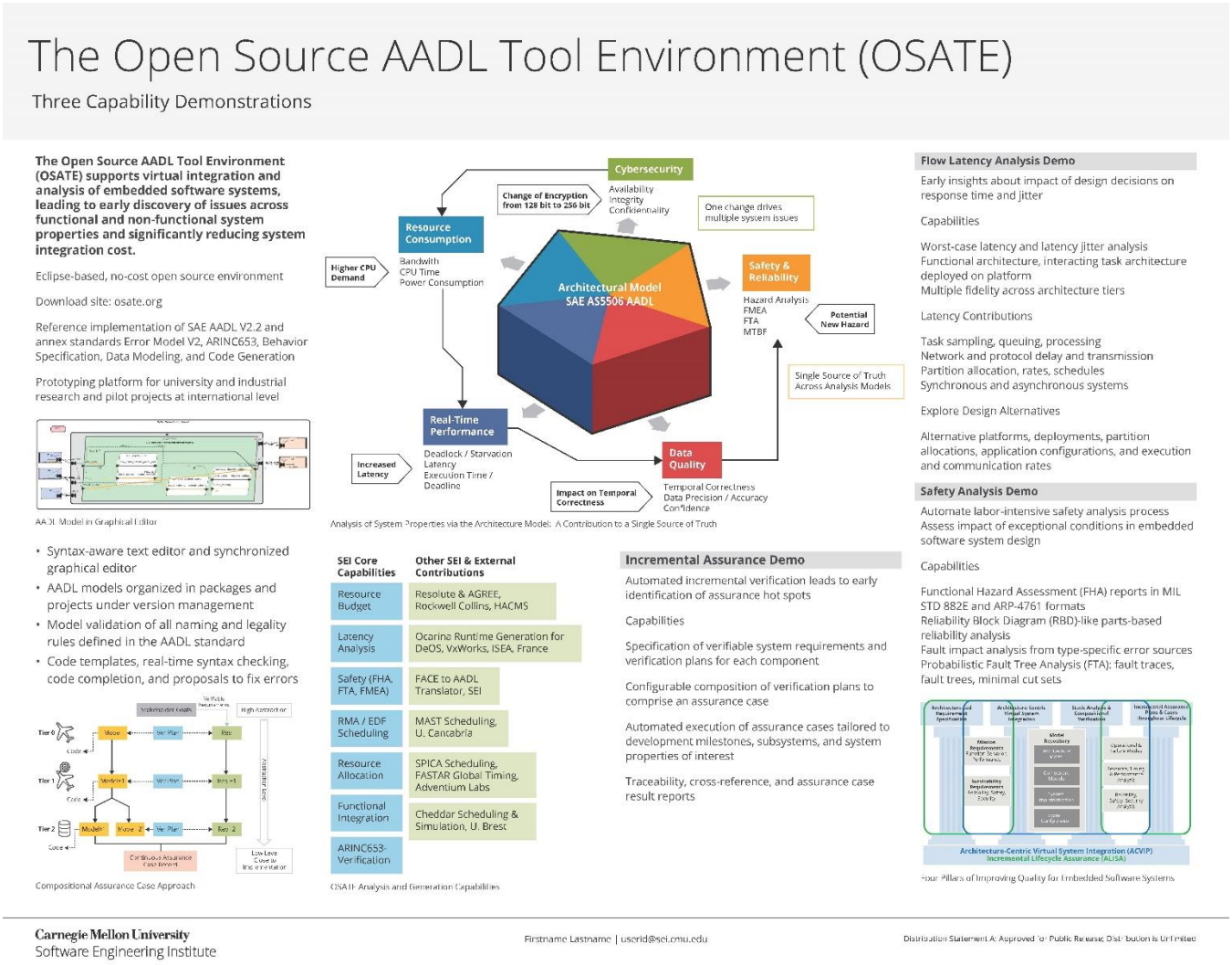
Each analysis produces reports in multiple formats—such as Excel and CSV—that facilitate discovery of potential issues and help designers build their architectures.

An Emphasis on Safety-Critical Systems

OSATE supports the Error Model Annex of AADL for specifying a system's fault behavior in the architecture model. Engineers can specify error occurrence and propagation in their architectures using the textual notation of AADL. OSATE includes several functions for processing this information and generating validation materials required by validation standards, such as

- Functional Hazard Assessment: description of faults that occur in each system function
- Fault-Tree Analysis: occurrence of hierarchical dependencies between faults within the architecture. OSATE integrates its own Fault-Tree Analysis tool, EMFTA.

These tools have been evaluated and designed to support industrial practices, such as the SAE ARP4761 standard.



B.3.4 Rhapsody

(See [Engineering Systems Design Rhapsody - Overview | IBM])

IBM® Engineering Systems Design Rhapsody® (Rational Rhapsody) and its family of products offers a proven solution for modeling and systems design activities that allows you to manage the complexity many organizations face with product and systems development. Rhapsody is part of the IBM Engineering portfolio that provides a collaborative design development, and test environment for systems engineers that supports UML, SysML, UAF as well as AUTOSAR import and export capabilities. The solution also allows for control of defense frameworks (DoDAF, MODAF and UPDM) and helps accelerate industry standards such as DO-178, DO-178B/C and ISO 26262.

Key Features:

- Analyze and elaborate project requirements
- Rapidly move from design to implementation
- Automate design reviews and generate documentation
- Prototype, simulate and execute designs for early validation
- Work in an embedded, real-time agile engineering environment

B.3.5 CAMEO Enterprise Architecture (Cameo EA)

(See https://www.3ds.com/products-services/catia/products/no-magic/cameo-enterprise-architecture/)

CATIA No Magic has deep experience with DoDAF 2.0, MODAF, NAF 3 and UAF. Our Cameo Enterprise Architecture product, based on our core product MagicDraw, offers the most robust standards compliant DoDAF 2.0, MODAF, NAF 3, NAF 4, and UAF 1.1 via a UAF standardized solution. And what's more, CATIA No Magic fully supports all architectural framework products ensuring you achieve project results. CATIA No Magic also leads the industry in its integration in systems of systems engineering, ensuring that you achieve net-centric success. Meet your interoperability challenges with proven, tested No Magic solutions.

CATIA No Magic Specifically Meets DoDAF 2.0, MODAF, NAF 3 and UAF Needs.

Improved Project Results - Your team will do a better job of mining available data, measuring and visualizing architecture and overall success factors resulting in improved project results.

- Convey the knowledge faster and easier
- Easily represent and communicate complex architecture
- Reduce assumptions, misconceptions and risk

Program Accountability - Provide Program Manager accountability including the enablement of net-centric processes and architectures, flexibility and responsiveness.

- Meet standards and easily follow guidance
- Understand risk/cost
- Gaps are identified and eliminated

Resource Management - CATIA No Magic's MagicDraw solution greatly facilitates the efficient and effective deployment of IT resources. The tool automates and assists the process of resource allocation ensuring project critical project success.

- Use "what-if" scenarios to confirm and calculate project success criteria
- Quickly identify under or over utilized resources
- Are the resources deployed in a project?
- Ensure that resources are allocated to meet specific project goals

Success metrics - Metrics lead to program success. Employing MagicDraw in UAF promotes significant improvement in processes, program and people efficiencies as well as shorter cycle times. Rapidly see and identify data relationships and critical paths.

- Align your operational metrics with your system metrics
- Use "what-if" scenarios to optimize system and operational parameters
- Use metrics to drive system design and operations
- Trace metrics back to systems that fulfill them

Efficiency - Benefit from improved speed to deploy, optimized resource allocation, improved collaboration and reduced overall cycle times. Teams will find it easier to take the abstract and make it meaningful, and achieve net-centric results.

- Be efficient and collaborative
- Easily and quickly pass along knowledge and training to others

- Achieve efficiencies because guidance has been followed

B.3.6 Dynamic Object-Oriented Requirements System (DOORS)

(See Overview of Rational DOORS - IBM Documentation)

Rational® DOORS® is a leading requirements management tool that makes it easy to capture, trace, analyze, and manage changes to information. Control of requirements is key to reducing costs, increasing efficiency, and improving the quality of your products.

DOORS is an acronym for Dynamic Object-Oriented Requirements System. Using the Rational DOORS family of products, you can optimize requirements communication, collaboration, and verification throughout your organization and across your supply chain.

At the heart of the family is Rational DOORS, an application that runs on Windows, Linux, and Solaris systems. With its own built-in database, Rational DOORS provides a rich set of features to help you capture and manage requirements.

Rational DOORS makes it easy for everyone in your organization and beyond to participate in and contribute to the requirements management process:

- Using a web browser, you can access your requirements database through Rational DOORS Web Access.
- You can manage changes to requirements with either a simple predefined change proposal system or a more thorough, customizable change control workflow through integration to Rational change management solutions.
- With the Requirements Interchange Format, you can directly involve suppliers and development partners in the development process.
- You can link requirements to design items, test plans, test cases, and other requirements for easy and powerful traceability.
- Business users, marketing, suppliers, systems engineers, and business analysts can collaborate directly through requirements discussions.
- Your testers can link requirements to test cases using the Test Tracking Toolkit for manual test environments.
- You can use the Open Services for Lifecycle Collaboration (OSLC) specifications for requirements management, change management, and quality management to integrate with systems and software lifecycle tools.
- You can integrate with other Rational tools, including Rational Team Concert, Rational Quality Manager, Rational DOORS Next Generation, Rational Rhapsody®, Rational Focal Point™, Rational Insight, and Rational System Architect, and also many third-party tools, providing a comprehensive traceability solution.