AVIONICS SOFTWARE ENGINEERING

Requirements Management Guidebook

Software Requirements Management Working Group

30SEP98

This page left intentionally blank

RECORD OF CHANGES

A - ADDED **M** - MODIFIED **D** - DELETED

CHANGE NUMBER	DATE	NUMBER OF FIGURE, TABLE OR PARAGRAPH	**A M D**	TITLE OR BRIEF DESCRIPTION	CHANGE REQUEST NUMBER

This page left intentionally blank

FOREWORD

The Requirements Management Guidebook was developed at the request of Naval Air Systems Command through the Computer Resources Group's (CRG) Requirements Management Working Group (RMWG). Sponsorship has subsequently been assumed by the Software Engineering Competency Executive Leadership Team (SWELT). The Guidebook provides a framework for establishing and maintaining a requirements management discipline for software engineering activities and provides assistance in tailoring the process to meet project-specific goals. The Requirements Management process contained in this document meets the objectives of the Software Engineering Institute's (SEI) Key Process Area (KPA) for Requirements Management. In addition, the process has been expanded to allow application to systems engineering activities.

For additional copies or to submit comments, please contact:

Department of the Navy
Naval Air Systems Command
ATTN: Software Engineering Division (AIR 4.5.7)
22347 Cedar Point Road
Patuxent River, MD 20670

(301) 342-2103 Voice
(301) 342-2149 Fax

"guidoas.nimitz@navair.nav.mil" e-mail

Acknowledgments

The 31 October 1995 handbook would not have been possible without the continued support of the CRG. Special thanks to the following individuals who played a key role in the development of this first draft:

COMOPTEVFOR	*Jeff Gerlitz*
NAWC - Aircraft Division	*Dennis Baker; Luke Campbell; Frank Hollenbach; Peter McDevitt; Ruth Pickering*
NAWC - Weapons Division	*Steve Bullard; Susan Hiser (Distributed Information Systems, Inc.); Dan Moretti*
NAVAIR Headquarters	*David Vennemann; Doug Schaus (PRC, Inc.)*
NCCOSC	*George Robertson*

The 1 April 1996 guidebook would not have been possible without the continued support of the SWELT. Special thanks to the following individuals who played a key role in the development of the second draft:

COMOPTEVFOR	*Jeff Gerlitz*
NAWC - Aircraft Division	*Frank Hollenbach; Peter McDevitt*
NAWC - Weapons Division	*Steve Bullard; Dan Moretti*
NAVAIR Headquarters	*Doug Schaus (PRC, Inc.)*
NCCOSC	*George Robertson (OSE Corp.)*

The 30 September 1996 guidebook was made possible by continued support of the SWELT. A number of comments on earlier guidebook versions were reconciled and included. The following individuals were responsible for resolution of comments and updates to the document:

NAWC - Aircraft Division ..... Frank Hollenbach

NAWC - Weapons Division..... Steve Bullard; Dan Moretti

NAVAIR Headquarters ..... Tom Coyle; Doug Schaus (PRC, Inc.)

NCCOSC ..... Ann Hess; George Robertson (OSE Corp.)

The July 1997 guidebook was made possible by continued support of the SWELT. Comments on the previous guidebook were reconciled and included in that release. The following individuals were responsible for resolution of comments and updates to the document:

NAWC - Aircraft Division ..... Frank Hollenbach; Peter McDevitt

NAWC - Weapons Division..... Steve Bullard; Dan Moretti

NAVAIR Headquarters ..... Tom Coyle; Dee Howard (PRC, Inc.)

The 9 June 1998 guidebook was made possible by continued support of the SWELT. Comments on earlier guidebook versions have been reconciled and included in this release. The following individuals were responsible for resolution of comments and updates to the document:

NAWC - Aircraft Division ..... Peter McDevitt

NAWC - Weapons Division..... Dan Moretti

NAVAIR Headquarters ..... Tom Coyle; Ken Ptack (PRC, Inc.)

TABLE OF CONTENTS

TABLE OF FIGURES

1. INTRODUCTION

Inadequate, incomplete, erroneous, and ambiguous system and software requirements are a major and ongoing source of problems in systems development. These problems manifest themselves in missed schedules, budget overruns, and systems that are, to varying degrees, unresponsive to the true needs of the customer. These manifestations are often attributed to the poorly defined and ill-understood processes used to elicit, specify, analyze, manage, and validate requirements.

The process described in this document provides information and guidance to personnel involved in the management of system and software requirements. This guidance is intended to satisfy the process activities described for requirements management in the *Key Practices of the Capability Maturity Model, Version 1.1, February 1993.* The requirements management process contained in this guidebook provides guidance to the practitioners so that each project may tailor the process as is appropriate for the specific site and/or project.

This guidebook assumes that each project performs the tasks listed below.

- Implements requirements management beginning with project kickoff and continuing through project wrap-up.
- Assigns personnel for specific requirements management responsibilities.

1.1 Purpose

The purpose of this document is to describe the process activities common to all organizations obliged to perform requirements management. This document identifies and describes a requirements management process to achieve repeatability.

1.2 Scope

The requirements management process described in this document begins with the **Elicitation** of customer requirements and progresses through an incremental, evolutionary process providing additional clarification and definition of the requirements as the products are developed. The management of requirements by the developer is completed with delivery of the products that satisfy the acceptance criteria.

1.3 Document Overview

This document is intended to provide an overview of repeatable processes that personnel can use in providing requirements management support to a project.

This document is organized into the sections listed below.

- **Section 1** provides the scope and purpose of this document.
- **Section 2** lists the Government standards and other publications referenced in this document and used in its preparation.
- **Section 3** describes the requirements management process model and how it applies to the requirements management process contained in this guidebook.
- **Section 4** contains the requirements management process. It describes the inputs, participants, activities, products and processes of requirements management.

- **Appendix A** lists the terms and definitions used in this document. It also contains a description of the roles and responsibilities of those who participate in the requirements management process. To effectively use this document, the reader should first become familiar with the definitions section of this document.
- **Appendix B** contains process diagrams for the requirements management process described in Section 4.
- **Appendix C** contains examples of requirements management as applied to a classic DOD-STD-2167A development effort, as used by the F/A-18 program, and a description of one project's use of the Requirements and Traceability Management (RTM) tool in performing requirements management.
- **Appendix D** contains a process checklist for the process contained in Section 4.
- **Appendix E** provides suggestions for requirements management metric collection points.
- **Appendix F** provides a brief discussion of different development models that a particular project may want to utilize.
- **Appendix G** identifies many of the standards that govern requirements management activities, as well as the paragraphs in this document that address those topics.

Throughout the guidebook, words in boldface within the body of text are process activities that are defined with the definitions section. The information below provides the format and format definitions for describing a process activity for requirements management.

Participants: The individuals or groups that perform a function to accomplish the activity. Ordering of the participants is by the anticipated level of participation within the activity, although the amount of participation is recognized to be unique to the mix of organizations, projects and requirements level.

Entry Criteria: The elements and/or conditions necessary to be in place to begin an activity.

Input: Data or material with which an activity is performed.

Steps: Actions to transform an input into a predetermined output.

Exit Criteria: Elements and/or conditions necessary to be in place to complete an activity.

Output: Data or material produced by or resulting from an activity. It must include the input data in some form. The output title differs from the input title in order to indicate that an activity has been performed.

2. REFERENCED DOCUMENTS

This section lists the specifications, standards, manuals and other documents referenced or used as source material for this document.

2.1 Government Documents

Listed below are the Government documents used in the development of this handbook.

MIL-STD-490A	Military Standard Specification Practices, 4 June 1985
MIL-STD-499B	Systems Engineering, DRAFT
MIL-STD-498	Software Development and Documentation, December 1994
MIL-STD-1521B	Technical Reviews and Audits for System, Equipment and Computer Software (Notice 1), 19 December 1985
DOD-STD-1679A	Military Standard Software Development, October 1983
DOD-STD-2167A	Defense System Software Development, 4 June 1985
MIL-M-38784C	Military Specifications-Manuals, Technical: General Style and Format Requirements, 12 October 1990
DODI 5000.2-M	Defense Acquisition Management Policies and Procedures, February 1991
OPNAVINST 5000.42D	Test and Evaluation Policy and Procedures
	Software Development Plan (SDP) for the ACMS Project
RL-TR-93-316	A Guide to Total Software Quality Control, Griffis Air Force Base NY, Rome Laboratory, Air Force Material Command, Clapp Judith A Stanten, Saul F., December 1992

2.2 Non-Government Documents

Listed below are the non-government documents used in the development of this handbook.

CMU/SEI-93-TR-25	Key Practices of the Capability Maturity Model, Version 1.1, February 1993, Software Engineering Institute
IEEE-STD-610.12	IEEE Standard Glossary of Software Engineering Terminology, Approved September 1990, IEEE Standards Board
IEEE-STD-1061-1992	IEEE Standard for a Software Quality Metrics Methodology, 1992
IEEE 610.12-1990	IEEE Standard Glossary of Software Engineering Terminology.
IEEE 1220-1994	IEEE Trial-Use Standard for Application and Management of the Systems Engineering Process, February 28, 1995, IEEE Computer Society.
ISO 9000	ISO Quality Standards Collection (A Global Compilation), Global Professional Publications
ISO/IEC 9126	Information Technology - Software Product Evaluation Software Product Evaluation Quality Characteristics and Guidelines for Use, Revision E, 1991
	A Spiral Model of Software Development and Enhancement, Barry W. Boehm, TRW Defense Systems Group, IEEE Computer, May 1988.

3. PROCESS MODEL OVERVIEW

The requirements engineering process involves taking broad and abstract mission needs obtained from the customer and decomposing them into precise, unambiguous requirements that can be easily understood by the software engineers writing code and the hardware engineers laying out circuit boards. The requirements management process involves setting up a documented, repeatable process in which requirements engineering can occur.

This section identifies several requirements management process models that can be used to decompose requirements into a series of more focused, manageable pieces that can be implemented by software and hardware engineers. As the pieces are decomposed and become more focused, the requirements become more detailed. Section 4 describes a Requirements Management process that can be used during each step of the chosen requirements management model to restate abstract requirements in less ambiguous terms.

The three common requirements management process models are spiral, waterfall and iterative build. Each of these models has a number of variations. Additional information on requirements process models is available in Appendix X.

The requirements model chosen depends on the specific non-technical requirements of the project. The iterative build model might prove a good choice for a project with unproven technology or a project with many user interactions because it provides a mechanism for fielding a working prototype sooner in the development process. The spiral model provides a mechanism for supporting projects with many requirements that can be broken down into separate subcomponents. The classic waterfall model can be used on a very simple project with a minimal requirements engineering task. Two or more models can be merged together to create a successful hybrid model.

When choosing a requirements management model, consider the following two points.

- A successful model must provide a mechanism for going backwards in the process as defects in previously defined and documented requirements are identified. The requirements management and engineering activities associated with capturing the defective requirement must be repeated. In addition, the requirements decomposition activities that occurred between the point where the defect was fixed and where it was identified must be repeated. This will ensure that the consequences of the changed requirement are fully assessed. In the spiral model, this can be achieved using subspirals.
- A successful requirements management model should also provide a mechanism for conducting requirements engineering activities in parallel. In a large project, at some point the requirements will be refined to the point where they can be assigned to specific groups (e.g. software, hardware, subsystem) who can then continue to decompose the requirements to the point which the requirements are sufficiently detailed to be implemented. This concept is also represented in the spiral model with subspirals.

This page left intentionally blank

4. REQUIREMENTS MANAGEMENT PROCESS

The process described in this section presents a requirements management framework that supports the items below.

- An evolutionary, incremental approach to requirements development.
- A team approach to requirements development.
- A disciplined approach to requirements development.
- Ensuring that changes in requirements are evaluated, meet the quality attributes and are reflected in changes to the formalized engineering artifacts and traceability work products.
- In addition, the framework supports the capturing of decisions and rationale.

This process is intended to be an iterative process that is performed at various levels of a system decomposition (e.g., platform, system, subsystem, Weapon Replaceable Assembly (WRA), component). As such, the roles of the participants and acceptance criteria may vary with each cycle. For example, at the highest level the customer may be the Program Executive Officer (PEO) and at subsequent requirements management cycles, the customer may be a Program Manager Air (PMA) or Integrated Product Team (IPT). Figure 4-1 illustrates the requirements management sectors, and 6-1 presents the overall flow of activities and participation for the requirements management process.

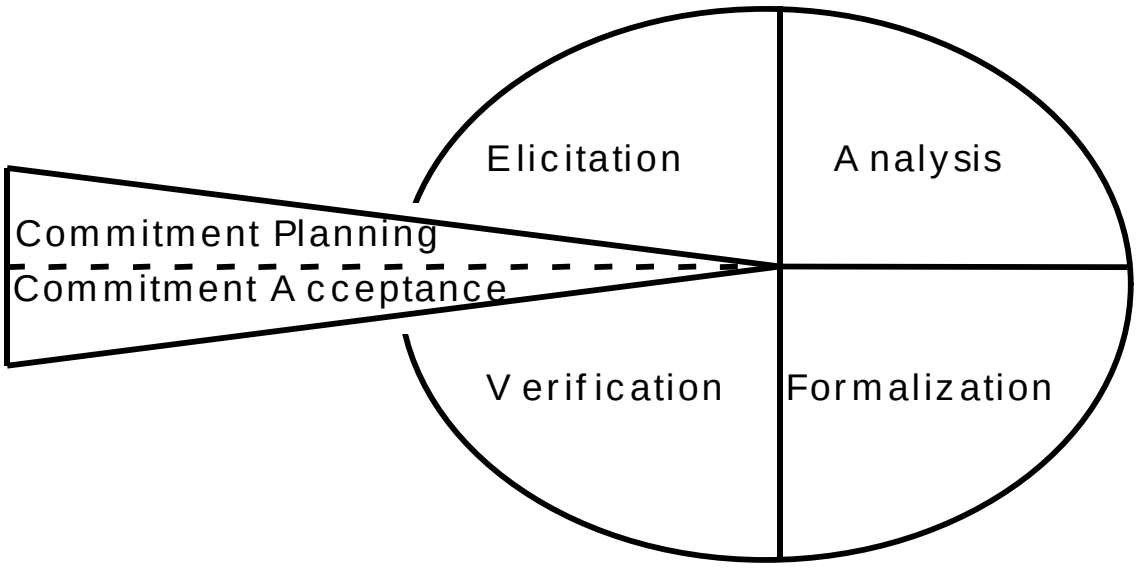


Figure 4-1. Requirements Management Sectors.

Participants - Senior Management, Customer, Project Management, Systems Engineering, Software Engineering, CM/QA, Functional Testing, Operational Testing, End User

Entry Criteria - The need for planning and agreement of new or changed requirements has been identified.

Inputs - Mission Need, Operational Scenario, Requirements, Change or Deficiency Report

Steps:

(1) **Commitment Planning** - During this activity the goals, constraints and acceptance criteria for this spiral cycle are agreed upon by the stakeholders. Refer to Paragraph 4.1 for a description of the **Commitment Planning** activity.

(2) **Elicitation** - An **Elicitation** of customer requirements is performed to obtain an understanding of the operational scenario, mission needs, or changes documented in a change or deficiency report. Refer to Paragraph 4.2 for a description of the **Elicitation** activity.

(3) **Analysis** - An **Analysis** of the requirements is performed to evaluate the quality of the requirements. Refer to Paragraph 4.3 for a description of the **Analysis** activity.

(4) **Formalization** - A team understanding of the requirements, their traceability and supporting rationale are recorded or documented in the **Formalization** activity. Refer to Paragraph 4.4 for a description of the **Formalization** activity.

(5) **Verification** - The **Verification** activity is used to review, modify (if necessary) and baseline the documented requirements. Refer to Paragraph 4.5 for a description of the **Verification** activity.

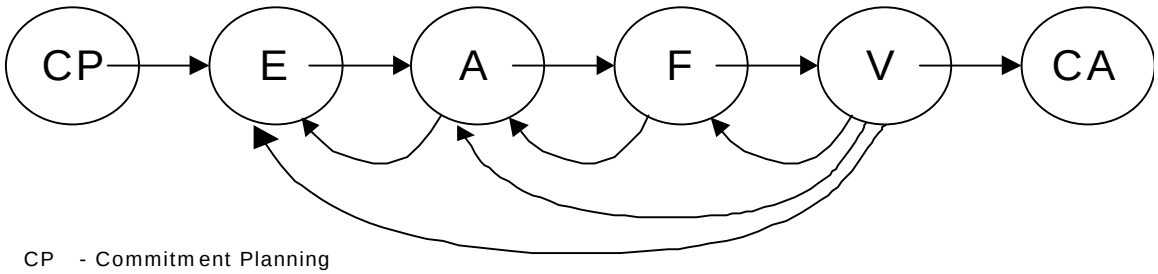
(6) **Commitment Acceptance** - The **Commitment Acceptance** activity provides stakeholders visibility into the current state of the process and products. Approval to proceed to the next requirements management spiral cycle or next process is obtained from the customer. Authenticated Requirements are produced. Refer to Paragraph 4.6 for a description of the **Commitment Acceptance** activity.

Outputs - Authenticated Requirements, Approval to Proceed

Exit Criteria - The customer has approved the baselined requirements as meeting the acceptance criteria for the cycle. Approval to proceed to the next process is obtained.

Activity Interactions

Interactions are present between neighboring activities during progression through a cycle. In addition, iterations that address preceding functional activities may be required to complete the current cycle. The interactions are determined by the nature of the defect or clarification that is required. For example, Figure 4-2 depicts elements discovered during the **Verification** that could be addressed in the **Elicitation**, **Analysis** and/or **Formalization** activities. In addition, Figure 4.2 suggests iterations between the **Analysis** and **Elicitation** activities, as well as between the **Formalization** and **Analysis** activities.



CP - Commitment Planning
E - Elicitation
A - Analysis
F - Formalization
V - Verification
CA - Commitment Acceptance

Figure 4-2. Activity Interactions.

4.1 Commitment Planning

The purpose of the **Commitment Planning** activity is to gain agreement among the stakeholders on what is to be done, who should do what, and by when things should be accomplished during the next evolution of the requirements development. This planning activity also includes reaching agreement on the acceptance criteria for the product. **Commitment Planning** provides the basis for the daily conduct of business. Figure 6-2 illustrates this process and participation.

NOTE:

Whenever a change or a deficiency report is addressed, all previous Requirements Management Process Model (RMPM) iterations associated with the change or deficiency report must be evaluated. This means to completely implement a single change, multiple RMPM iterations may be necessary to address each level of requirements definition.

Participants - Senior Management, Customer, Project Management, Systems Engineering, Software Engineering, CM/QA, Functional Testing, Operational Testing, End User

Entry Criteria - The need for planning and agreement on the implementation of new or changed requirements has been identified.

Input - Mission Need, Operational Scenario, Requirements, Change or Deficiency Report, Baselined Requirements, Verification Reports

Steps:

(1) **Identify Stakeholders** - The individuals and/or groups with a stake in the success of the project are identified and asked to assist in the refinement of the technical and non-technical requirements. Generally, this includes obtaining Points Of Contact (POC) from senior management, product teams, and competencies and obtaining necessary communication information (e.g., e-mail address, phone and FAX numbers). Each cycle may require that the stakeholders be expanded to include additional members with the skills necessary to complete this cycle.

(2) **Define Acceptance Criteria for this Cycle** - The stakeholders reach and document agreement on the objectives for this cycle of the requirements management spiral. For example, determining and documenting all system-level requirements may be the objective of one pass through the spiral. The acceptance criteria may be that the system-level requirements and documentation are completed and delivered for customer review by a given date.

(3) **Identify Non-technical Requirements** - Non-technical requirements, which may not be documented to this point, are identified, documented, and understood. These requirements may include the items listed below.

- Products to be delivered
- Delivery dates
- Milestones
- Programmatic constraints
- Facility availability
- Identify development model (e.g., Waterfall, Iterative Build, Spiral)

(4) **Develop/Update Plan** - A plan is documented which includes resource estimates, a schedule of activities and milestones, and a cost estimate for continuing efforts. The affected groups provide inputs to the plan. During each pass through the spiral, changes in the non-technical requirements may require modifications to the plan. Modifications to inputs provided by the affected groups are not implemented unless they are negotiated and agreed to by the affected

groups. The Project Planning and Project Tracking & Oversight processes typically handle development and update of the plan.

(5) **Assess Project Risk** - The risk of the project should be continually assessed to ensure that **Commitment** to the plan represents a reasonable risk. Unreasonable risks should be clearly documented in the plan. Changes to requirements may impact the risk associated with a development effort. Risk should be reassessed each time requirements are evaluated for incorporation into the project.

NOTE:

"Reasonable" risk may vary greatly from project to project; an unreasonable risk for a Fleet product may be quite reasonable for a research activity.

(6) **Assign/Review Requirements Management Policy and Responsibilities** - The project management develops or reviews a written organizational policy for managing requirements throughout the project life cycle. This policy explicitly assigns responsibility for the management, analysis and documentation of requirements and their allocation, as well as changes to requirements. Trained personnel (including system/software engineering, and affected groups), tools, resources and funding are identified and agreed as adequate for performing requirements management. Requirements measurements are identified and agreed upon by stakeholders. Refer to Appendix B for suggested data collection types. The policy and responsibilities should be documented in a plan.

(7) **Review Plan** - All affected groups review the plan to reach concurrence on the contents of the plan, with particular emphasis on cost, schedule, resources and risk. The affected groups should ensure that they have, or can acquire, the resources required to complete the project. In effect, concurrence (agreement) with the plan indicates commitment by the affected groups. This agreement should be documented.

(8) **Review by Senior Management** - Senior management should review the plan. If the plan is found to be acceptable, then the plan can be approved. The commitment made by senior management through the approval of the plan should be documented.

(9) **Show Commitment to the Plan** - The funding and resources necessary for requirements management have been identified and allocated.

Output - Mission Need, Changed Mission Need, Requirements, Changed Requirements, Approved Plan

Exit Criteria - Agreement has been reached and documented among the stakeholders on what is to be done, by whom, and when. Resources and funding for requirements management have been identified and allocated. Personnel have been trained or are skilled in the necessary aspects of requirements management and the application domain. Requirements management responsibility has been assigned. Approved plan is produced and made available to participants.

4.2 Elicitation

The purpose of the **Elicitation** activity is to gather information from the customers (e.g., issues, needs, higher-level requirements) and capture the information as candidate requirements (both technical and non-technical) along with the supporting rationale for each requirement.

Changes to requirements and mission needs are inputs to **Elicitation**. These changes are transformed through this activity to candidate technical requirements, as well as non-technical requirements, and after exiting from this activity are no longer considered changes. Figure 6-3 illustrates this process and participation.

Participants - Customer, Project Management, Systems Engineering, Software Engineering, CM/QA, Functional Testing, Operational Testing, End User

Entry Criteria - Agreement has been reached between the stakeholders on what is to be done, by whom, and when. Resources and funding for requirements management have been identified and allocated. Personnel have been trained or are skilled in the necessary aspects of requirements management and the application domain. Requirements management responsibility has been assigned.

Input - Mission Need, Changed Mission Need, Requirements, Changed Requirements, Approved Plan

Steps:

(1) **Conduct Fact-Finding** - Meeting, interviews, observation, or other methods are utilized to obtain a mutual understanding of the technical and/or non-technical requirements in accordance with the approved project plan. The purpose of this step is stated below.

- Determine the operational and problem context.
- Gather information and develop an understanding of open issues.
- Communicate open issues for resolution between the customer and other stakeholders.
- Derive requirements from higher-level requirements (as appropriate).
- Update stakeholder list (as required).

(2) **Capture Candidate Technical Requirements** - Mission needs, changed mission needs, requirements, and changed requirements are used as the source for candidate technical and non-technical requirements. The candidate technical requirements are captured along with their supporting rationale. This may be performed through a variety of techniques including those listed below.

- Group development techniques (e.g., Joint Application Design (JAD))
- Questionnaires
- Simulation
- Storyboarding
- Scenario development
- Prototyping

All levels of candidate requirements and their supporting rationale are retained for use later in the requirements management process. An arbitrary breakout may be performed to facilitate the collection of candidate requirements (e.g., functional areas, security, logistics, environment, or training).

At this point in the process each project should consider establishing a method (e.g., database, traceability matrix) to help ensure requirements can be traced. Traceability of the requirements will be required in the **Formalization** activity.

(3) **Capture Non-technical Requirements** - The requirements that impact the system/software and/or project management's ability to meet the acceptance criteria are retained for planning, estimating, and resource oversight. Capture rationale for non-technical requirements. Examples of non-technical requirements that may impact project plans include the items listed below.

- Facility availability
- Programmatic constraints

Exit Criteria - An understanding of the candidate technical requirements has been obtained, requirements have been named, and the supporting rationale have been captured.

Output - Candidate Technical Requirements, Non-technical Requirements, Supporting Rationale

4.3 Analysis

The purpose of the **Analysis** activity is to transform candidate technical requirements into informal requirements by ensuring that they reflect the quality attributes of requirements and express the needs of the customer.

Analysis is an iterative activity. The process steps will likely have to be repeated several times, with consultation between customers, end users and developers during each iteration, and communication throughout the steps. Figure 6-4 illustrates this process and participation.

Participants - Project Management, Systems Engineering, Software Engineering

Entry Criteria - An understanding of the candidate technical requirements has been obtained along with the supporting rationale.

Input - Candidate Requirements, Non-technical Requirements, Supporting Rationale

Steps:

NOTE:

The sequence of these steps is determined by each project and may differ from the order stated below.

(1) **Categorize** - Requirements are decomposed into general categories and cohesive units. This can be along functional lines or can be along performance lines. Examples of possible classifications are listed below.

- Classification (e.g., flight safety, functional area, state-of-requirement, security)
- Allocation (e.g., platform, system, hardware, software, subsystem, component)

(2) **Determine Quality Attributes for Each Requirement** - Each requirement is reviewed against a set of project-defined attributes. Requirements that do not reflect the necessary attributes require further refinement (e.g., investigation, communication with customer). Several of the quality attributes are listed in the Appendix A, Section 5.2.

(3) **Establish Traceability** - Each requirement is analyzed to ensure that it traces back to a stated need, goal or higher-level requirement.

(4) **Reconcile Requirements** - Requirements reconciliation is a means of ensuring that the system's stated requirements are a complete and correct reflection of the customer's needs. This may be done by employing prototyping, modeling, and simulation (or some combination thereof). A few examples of methods used to reconcile requirements are listed below.

- Prototyping is a technique used to demonstrate the desired functionality or behavior of a proposed system. A prototype is a tangible, physical model, or a Graphical-User Interface (GUI)-based software model. While a physical hardware prototype may not be practical due to the immaturity of the requirements, a software prototype can be feasible and beneficial. Its primary goal is to serve as a learning vehicle to provide more precise ideas about the system under development. Often focusing on the Human-Machine Interface (HMI), a prototype aids in determining completeness of requirements because the customer and/or end user can interact with the prototype and decide whether it reflects all needs and goals.
- Modeling involves the use of software to create a computer model of the system under development. The model may incorporate one or more views of the system: architectural, functional, behavioral, or performance. Modeling is related to prototyping but does not necessarily feature an HMI mock-up, such as the case where the system under development is a "black box" subsystem, which does not interface with a human operator.

- Simulation brings prototype and models to life by animating the actual execution of the system under development. An end-user can "operate" a simulated prototype of an HMI such as a cockpit display. A model of a black box can be simulated to show how it will process input and provide output. Simulation of a prototype or model aids in determining correctness of requirements because it exhibits erroneous behavior if any of the modeled requirements have not been defined correctly.

(5) **Capture Decision Rationale** - Record the rationale for any decisions and retain for use in this and future projects. Trade studies may be performed to record the rationale.

Exit Criteria - Informal Requirements have been produced through the assessment of candidate requirements, categorization of requirements, establishment of traceability, reconciliation and capturing of decision rationale.

Output - Informal Requirements, Requirements Traceability

4.4 Formalization

The purpose of **Formalization** is to transform informal requirements data into a form that clearly communicates the desired system/software product. The result of the **Formalization** activity is a vehicle for conveying the requirements among all stakeholders. At this point in the requirements management process, the configuration control and the level of control are determined by the project. Configuration control of the requirements is also initiated. Figure 6-5 illustrates this process and participation.

Many formalized requirements/formalized engineering artifacts are documents whose content conforms to such standards as MIL-STD-490B, DOD-STD-2167A, MIL-STD-1679A, MIL-STD-498, and IEEE 1498.

Participants - Systems Engineering, Software Engineering, CM/QA

Entry Criteria -The informal requirements have been produced through the assessment of candidate requirements, categorization of requirements, establishment of traceability, reconciliation and capturing of decision rationale.

Input - Informal Requirements, Requirements Traceability

Steps:

(1) **Transform Requirements to Formal Engineering Artifact** - Obtain and document each informal requirement including the supporting information used in its development.

The informal requirements are documented in the agreed upon media and format. The desired media and format could be paper documents, digital documents, Computer-Aided Software Engineering (CASE) tool output, databases, models and/or prototypes.

(2) **Formalize Traceability of Formal Engineering Artifact** - Document the traceability between the formalized requirement/formalized engineering artifact and its associated source in a maintainable format or traceability work product.

(3) **Place Formal Engineering Artifact Under Configuration Control** - Place the formal requirement/formal engineering artifact and formal traceability work product under CM.

Output - Formal Requirements, Formal Engineering Artifact, Formal Traceability Work Product

Exit Criteria - The formal requirements/formal engineering artifact and formal traceability work products have been placed under CM.

4.5 Verification

The purpose of the **Verification** activity is to provide assurance that the formal engineering artifacts are truly representative of the customer and end user system/software requirements. This is accomplished by presenting the formal engineering artifacts to the stakeholders, actively involving the stakeholders with the formal engineering artifacts and facilitating agreement among the stakeholders. Figure 6-6 illustrates this process and participation.

Participants - Customer, Project Management, Systems Engineering, Software Engineering, CM/QA, Functional Testing, Operational Testing, End User

Entry Criteria - The formal requirements/formal engineering artifacts and formal traceability work products have been placed under CM.

Input - Formal Requirements, Formal Engineering Artifact, Formal Traceability Work Products, Operational Scenario

Steps:

(1) **Perform Assessment** - An assessment of the documented requirements is performed to ensure that the topics listed below have been addressed.

- Inspect each documented requirement for the elements stated below.
  - Verify quality attributes have been addressed in the preparation of the requirement.
  - Uncover inconsistencies among requirements.
  - Identify redundancies among requirements.
  - Ensure no unintended changes to previous baselines have occurred.
- Identify impact of derived requirements on the operational concept/scenario.
- Verify that candidate technical requirements have been incorporated in the Formal Traceability Work Product.

(2) **Verify Traceability** - Listed below are methods for ensuring that traceability to the requirements exists.

- Verify that all formalized requirements are traceable to a higher-level requirement.
- Verify that all higher-level requirements are traceable to a formalized requirement(s).
- Identify any requirement without a parent.

(3) **Document Findings** - Findings are documented in a Verification or Deficiency Report(s).

- Create reports. These reports may contain unique identifiers for the report, types of engineering artifacts and requirements needing further definition.
- Place reports under CM.

(4) **Assign Deficiency for Resolution (If Required)** - Identify the activity where the investigation of the deficiency needs to be initiated.

(5) **Facilitate Agreement** - The formalized engineering artifact is used as a basis for establishing agreement between project management and the customer.

- The successfully evaluated formalized engineering artifact (may include portions that have been reworked or further refined) are examined by the stakeholders.
- The customer and stakeholders agree that the formalized engineering artifact is truly representative of the customer and end user requirements.

(6) **Establish a Requirements Baseline** - The formalized engineering artifacts are placed under CM to establish baselined requirements.

Output - Baselined Requirements, Verification Report or Deficiency Report

Exit Criteria - The formalized engineering artifacts have been reviewed, approved and placed under CM. Any deficiency reports result in a return to an activity for investigation.

4.6 Commitment Acceptance

The purpose of the **Commitment Acceptance** activity is to approve the baselined requirements as reflecting the acceptance criteria. This activity also provides a method of providing project and requirement status to the customer, project and senior management. Figure 6-7 illustrates this process and participation.

Participants - Senior Management, Customer, Project Management

Entry Criteria - The formalized engineering artifacts have been reviewed, corrected (as necessary), approved and placed under CM.

Input - Baselined Requirement, Verification Report

Steps:

(1) **Present Evidence of Baselined Requirement** - Project and requirements measurements collected to this point are presented to provide evidence and visibility to the current state of the project and requirements. Present the formalized engineering artifacts and formalized traceability work product for the baselined requirements.

(2) **Present Project Status** - Present plans, measurements and current status of the project including status of the baselined requirements according to a documented procedure. Review cycle objectives identified during **Commitment Planning**.

(3) **Formalize Commitment by All Stakeholders to Baselined Requirements** - (technical and non-technical) - The customer approval of the baselined requirements is obtained.

(4) **Obtain Approval to Proceed to the Next Process** - This approval may be to proceed to the next spiral of the requirements evolution or to the developmental engineering activity.

Output - Authenticated Requirements, Approval to Proceed

Exit Criteria - The customer has approved the baselined requirements as meeting the acceptance criteria. Approval to proceed to the next process is obtained.

5. APPENDIX A - GLOSSARY OF TERMS

The terms, definitions and roles listed below are provided as an aid in understanding the application of requirements management principles and processes.

5.1 Acronyms And Abbreviations

Listed below are the acronyms and abbreviations used in this document.

ACMS	Automated Communications Management System
A-Spec	A-Level Specification
ASAP	Aircrew Systems Advisory Panel
CASE	Computer-Aided Software Engineering
CDRL	Contract Data Requirements List
CI	Configuration Item
CM	Configuration Management
CMM	Capability Maturity Model
CMU	Carnegie Mellon University
COEA	Cost and Operational Effectiveness Analysis
COMOPTEVFOR	Commander, Operational Test and Evaluation Force
COTS	Commercial Off-the-Shelf
CRG	Computer Resources Group
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
CSU	Computer Software Unit
DBDD	Data Base Design Document
DCR	Document Change Request
DID	Data Item Description
DOD	Department of Defense
DODI	Department of Defense Instruction
FQT	Formal Qualification Testing
FRD	Functional Requirements Document
GPS	Global Positioning System
GUI	Graphical-User Interface
HMI	Human-Machine Interface
HWCI	Hardware Configuration Item
ID	Identification
IDD	Interface Design Document
IEEE	Institute of Electrical and Electronics Engineers, Inc.
I/O	Input/Output

IPT	Integrated Product Team
IRS	Interface Requirements Specification
IV&V	Independent Verification and Validation
JAD	Joint Application Design
KPA	Key Process Area
LGB	Laser-Guided Bomb
MIL	Military
MNS	Mission Needs Statement
MPE	Mission Planning Element
NDI	Non-developmental Item
OPNAVINST	Operations and Naval Instructions
ORD	Operational Requirements Document
PEO	Program Executive Officer
PMA	Program Manager, Air
POC	Point of Contact
POSIX	Portable Operating System Interface for Computer Environments
QA	Quality Assurance
RMPM	Requirements Management Process Model
RM Team	Requirements Management Team
RMWG	Requirements Management Working Group
RSM	Requirements Spiral Model
RTM	Requirements and Traceability Management
SDD	Software Design Document
SDP	Software Development Plan
SDR	System Design Review
SEG	Software Engineering Group
SEI	Software Engineering Institute
SOF	Statement of Functionality
SOR	Statement of Requirements
SOW	Statement of Work
SPS	Software Product Specification
SQA	Software Quality Assurance
SQL	Structured Query Language
SRR	System Requirements Review
SRS	Software Requirements Specification
SSDD	System/Segment Design Document

SSR	Software Specification Review
SSS	System/Segment Specification
STD	Standard
STD	Software Test Description
StP	Software through Pictures
TEMP	Test and Evaluation Master Plan
WBS	Work Breakdown Structure
WRA	Weapon Replaceable Assembly

5.2 Definitions

Listed below are definitions for the terms used in this document.

Acceptance Criteria - The criteria that a system or component must satisfy in order to be accepted by an end user, customer, or other authorized entity. [SEI/CMU-93-TR-25]

Activity - Any step or function performed, both mental and physical, toward achieving some objective. Activities include all the work the managers and technical staff do to perform the tasks of the project and organization. [based on 'activity' CMU/SEI-93-TR-25]

Approval to Proceed - Project management and the customer have agreed upon the goals of the next phase and have authorized work to continue. [Requirements Management Working Group (RMWG)]

Approved Plan **-** The software development and/or other plans that have been reviewed, approved, and committed to by project management, customers and affected individuals and/or groups. [IEEE-STD-1061]

Authenticated Requirements - The requirements that result from the **Commitment Acceptance** activity of the requirements management process. Baseline requirements that have been committed to by all stakeholders become authenticated requirements. [RMWG]

Baseline Requirements - The requirements that result from the **Verification** activity and are inputs to the **Commitment** activity of the requirements management process. [RMWG]

Candidate Technical Requirements - The initial customer requirements that have not been evaluated for any of the quality factors. The technical requirements that are captured during the **Elicitation** activity and are inputs to the **Analysis** activity of the requirements management process. [RMWG]

Changed Operational Need - A modification to a previously defined (and possibly) documented operational need. [RMWG]

Clarity - Requires that the requirement be readily understandable without semantic analysis. [RMWG]

Commitment - A pact that is freely assumed, visible, and expected to be kept by all parties. [CMU/SEI-93-TR-25]

Completeness - Requires that the requirement contains all of the information necessary, including constraints and conditions, to enable the requirement to be implemented such that the need will be satisfied. [RMWG]

Connectivity - Refers to the property whereby all of the terms within the requirement are adequately linked to other requirements and to word and term definitions, thus causing the individual requirement to properly relate to other requirements as a set. [RMWG]

Consistency - The degree of uniformity, standardization, and freedom from contradiction among the documents or parts of a system or components. [IEEE-STD-610.12-1990]

Correctness - (1) The degree to which a system or component is free from faults in its specification, design, and implementation. (2) The degree to which software, documentation or other items meet specified requirements. (3) The degree to which software, documentation or other items meet end user needs and expectations, whether specified or not. [IEEE-STD-610.12-1990]

Criticality - The degree of impact that a requirement, module, error, fault, failure or other item has on the development or operation of a system. [IEEE-STD-610.12-1990]

Customer - The individual or organization that is responsible for accepting the product and authorizing payment to the developing organization. [CMU/SEI-93-TR-25]

Deficiency Report - The written record of problems, errors or questions found within a product. This record is used to ensure resolution of the deficiency. [RMWG]

Developer - The organization tasked with developing the system/software and its associated technical data. [RMWG]

Elicitation - The process of gaining an understanding of customer and end user requirements, both technical and non-technical. [RMWG]

End User - The individual or group who will use the system for its intended operational use when it is deployed in its environment. [CMU/SEI-93-TR-25]

Engineering Artifact - A work product that documents or describes a design or operational aspect of a system at a given point in time. In requirements management, an engineering artifact is an open-ended representation that is used by end users and developers to explore, convey, or clarify a system condition or capability. [RMWG]

Feasibility - The degree to which the requirements, design, or plans for a system can be implemented under existing constraints. [IEEE-STD-610.12-1990]

Formal Engineering Artifact - A work product developed in accordance with a contract or governing obligation that documents or describes a design or operational aspect of a system. In requirements management, a formal engineering artifact is a system representation that satisfies a contract, standard, specification, or other formally imposed document and signifies an understanding or agreement between the end used and developer of the desired system conditions or capabilities. [RMWG]

Formal Requirements - The requirements and supporting rationale that have been evaluated, analyzed for the quality factors, documented (regardless of the media or form), and placed under configuration control. The formal requirements are used as the basis for software engineering and management activities. [RMWG]

Formal Traceability of Requirements - The established and controlled documentation (regardless of media or form) that shows direct and indirect relationships from the highest level of requirements to the lowest level of requirements. [RMWG]

Informal Requirements - The technical and/or non-technical requirements that have satisfied the quality factor evaluation but have not yet been formalized. [RMWG]

Modifiable - Refers to the property whereby all items within a requirement can be altered to meet future system/software needs. [RMWG]

Modularity - Requires that necessary changes to a requirement can be made completely and consistently and that the same requirement is specified only once. [RMWG]

Non-ambiguity - Requires that there be only one semantic interpretation of the requirement. [RMWG]

Non-technical Requirements - Agreements, conditions, and/or contractual terms that affect and determine the management activities of a system/software project (e.g., military standards, plans). [CMU/SEI-93-TR-25]

Operational Need - (1) The statements that identify the essential capabilities (the process or series of actions performed to effect a purpose or result) that are desired (or result) in the system under

development. [based on 'operational requirements' IEEE P1220 IEEE Standards for Systems Engineering] (2) The description of the overall desired characteristics from an operational viewpoint. It defines the general system goals, missions, functions and components of the desired system. [RMWG]

Plan - A document (e.g., Software Development Plan (SDP)) that describes the scheme for accomplishing defined objectives or work within specified resources. [based on 'plan standard' IEEE-STD-610.12-1990]

Process - A sequence of steps performed for a given purpose (e.g., the software development process). [based on 'process' IEEE-STD-610]

Project - An undertaking requiring concerted effort, which is focused on developing and/or maintaining a specific product. The product may include hardware, software, and other components. Typically a project has its own funding, cost accounting and delivery schedule. [CMU/SEI-93-TR-25]

Project Software Manager - The role with total responsibility for all the software activities for a project. The project software manager is the individual the project manager deals with in terms of software commitments and who controls all the software resources for a project. [CMU/SEI-93-TR-25]

Project Management - The role with total business responsibility for an entire project; the individual who directs, controls, administers and regulates a project building a hardware and/or software system. The project manager is the individual ultimately responsible to the customer. [based on 'project manager' CMU/SEI-93-TR-25]

Project Plan - Refer to definition of "plan."

Quality Attributes - The elements of a requirement that are necessary. Several different sets of quality attributes of requirements exist as identified below:

(1) Achievable, Feasible, Stable, Correct, Testable, Modifiable, Consistent, Non-ambiguous, Clear, Complete, Singular [RMWG]

(2) Efficiency, Functionality, Maintainability, Portability, Reliability, Usability [IEEE 1062]

(3) Functionality, Reliability, Usability, Efficiency, Maintainability, Portability [ISO 9126]

(4) Efficiency, Reliability, Usability, Maintainability, Expandability, Interoperability, Reusability, Integrity, Survivability, Correctness, Verifiability, Flexibility, Portability [Clapp 1992]

Repeatability - Basic project management processes are in place to track cost, schedule and functionality. The necessary process discipline is in place to repeat earlier successes on projects with similar applications. [CMU/SEI-93-TR-25]

Requirement - (1) A condition or capability needed by an end user to solve a problem or achieve an objective. (2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents. (3) A documented representation of a condition or capability as in (1) or (2). [IEEE-STD-610.12-1990] Requirements are both technical and non-technical proceeding through the following stages of maturity: Candidate, Informal, Formal, Baseline, and Authenticated. [RMWG]

Requirements Management - Refers to establishing and maintaining an agreement with the customer on the requirements for the system/software project. This agreement covers both the technical and non-technical requirements. This agreement forms the basis for estimating, planning, performing and tracking the software project's activities throughout the software life cycle. [based on purpose of 'Requirements Management' CMU/SEI-93-TR-25]

Requirements Traceability - The direct and indirect linkage of higher-level requirements to lower-level requirements implementation. The requirements are traceable through a document hierarchy beginning with the highest-level requirement (e.g., system) to the lowest-level requirement implementation (e.g., Computer Software Unit (CSU)). This results in the greatest number of requirements at the lowest level of the document hierarchy. [RMWG]

Senior Management - A management role at a high enough level in an organization that the primary focus is the long-term vitality of the organization, rather than short-term project and contractual concerns and pressures. In general, a senior manager for an organization would have responsibility for multiple projects. [based on "senior manager" CMU/SEI-93-TR-25]

Singularity - Refers to the attribute whereby a requirement cannot sensibly be expressed as two or more requirements having different subjects, verbs, and/or objects. [RMWG]

Stability - Refers to the likelihood of the requirement being modified or removed during the software development effort. [RMWG]

Stakeholders - The individuals who have interest in the success of a project. Examples of stakeholders include end users and the customer. Stakeholders may vary with each RMPM cycle and membership can expand or contract to include additional members with the skills necessary to successfully complete a cycle. [RMWG]

Supporting Rationale - The information necessary to ensure that an operational system or component fulfills its original requirements and any subsequent modifications to those requirements. [based on 'support' IEEE-STD-610.12-1990]

Technical Requirement - Those requirements that describe what the system or software must do and its operational constraints. Examples of technical requirements include functional, performance, interface and quality requirements. [CMU/SEI-93-TR-25]

Testability - (1) The degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met. (2) The degree to which a requirement is stated in terms that permit establishment of test criteria and performance of tests to determine whether the acceptance criteria have been satisfied. [CMU/SEI-93-TR-25] (3) Refers to the existence of a finite and objective process with which to verify that the requirement has been satisfied. [RMWG]

Traceability - The degree to which a relationship can be established between two or more requirements of the requirements management process, especially requirements having a predecessor-successor or master-subordinate relationship. [based on 'traceability' in the Dictionary of Computing Terms, IEEE 630-90, 1990]

Verification - The process of evaluating a system to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase. [based on 'verification' CMU/SEI-93-TR-25]

Verification Report - (1) The documentation (regardless of form or media) that states the success and/or failures of the system or software to perform in a manner consistent with the requirements. (2) The record that identifies the approval, action items, issues or problems found within a formalized engineer artifact through a review, assessment or inspection. [RMWG]

5.3 Roles And Responsibilities

The list below identifies the participants and their roles in the requirements management process and the responsibilities assigned to each.

Configuration Management (CM):

- Witnesses the **Commitment** process
- Manages baselined requirements

Customer (e.g., Sponsor, Designated Agents, Independent Verification and Validation (IV&V)):

- Defines requirements
- Reviews requirements
- Is actively involved in the **Commitment** process
- Participates in evaluating changes to requirements
- Participates in further definition and refinement of requirements

  Functional Testing
- Assists in the development of functional requirements definition

Operational Testing (e.g., Command Operation Test and Evaluation Test Force (COMOPTEVFOR):

- Assists in the development of operational requirements definition

End Users (e.g., identified Fleet staffs, End User):

- Assists in the development of functional requirements definition
- Reviews requirements
- Participates in evaluating changes to requirements
- Participates in further definition and refinement of requirements

Project Management activity:

- Reviews requirements
- Is actively involved in **Commitment** process
- Assures affected groups commit to requirements
- Approves requirements being incorporated in project

Quality Assurance (QA) activity:

- Reviews/Audits the activities and work products for managing the requirements and reports the results.
- Verifies that the software engineering group reviews the requirements and problems with the requirements are resolved before committing to them.
- Ensures that a requirements traceability matrix or similar tool is used to show that the product specifications cover the requirements.
- Ensures the software plans, work products and activities are appropriately revised when the requirements changes.
- Verifies that changes to commitments resulting from changes to the requirements are negotiated with the affected groups.
- Witnesses **Commitment** process

- Reviews report results

Senior Management:

- Reviews and Approves project plan
- Documents commitment to the project plan

Software Engineering activity:

- Is actively involved in **Commitment** process
- Analyzes requirements needing further definition or changes to ensure that the software requirements are met
- Reviews requirements
- Approves requirements being incorporated in project
- Participates in the evaluation and approval of changes to requirements
- Uses requirements as the basis for software plans and products

Systems Engineering activity:

- Is actively involved in **Commitment** process
- Analyzes requirements needing further definition or changes to ensure the system requirements are met
- Reviews requirements
- Approves requirements being incorporated in project
- Participates in the evaluation and approval of changes to requirements
- Uses requirements as the basis for system plans and products

Subcontractor Management:

- Reviews requirements
- Is aware of changes to requirements

6. APPENDIX B - PROCESS DIAGRAMS

This appendix contains diagrams (Figures 6-1 through 6-7) of the process described in Section 4. The diagrams identify participants for each of the activities.

Requirements Management

Participants: Senior Management, Customer, Project Management, Systems Engineering, Software Engineering, CM/QA, Functional Testing, Operational Testing, End User.

Entry Criteria: The need for planning and agreement of new or changed requirements has been identified.

Inputs	Processing	Outputs
Mission Need Operational Scenario Requirement Change or Deficiency	1. Commitment/Planning 2. Elicitation 3. Analysis 4. Formalization 5. Verification 6. Commitment/Acceptance	Authenticated Requirements Approval to Proceed

Exit Criteria: The customer has approved the baselined requirements as meeting the acceptance criteria for the cycle. Approval to proceed to the next process is obtained.

Figure 6-1. Requirements Management Processes.

Commitment/Planning

Participants: Senior Management, Customer, Project Management, Systems Engineering, Software Engineering, CM/QA, Functional Testing, Operational Testing, End User.

Entry Criteria: The need for planning and agreement on the implementation of new or changed requirements.

Inputs	Processing	Outputs
Mission Need Operational Scenario Change Request Baselined Requirements Verification Reports	1. Identify Stakeholders 2. Define Acceptance Criteria for this evolution of the spiral. 3. Identify Non-Technical Requirements 4. Develop/Update the Software Development Plan (SDP); this will include the cost/schedule impact. **5. Access Project Risk** 6. Assign/Review Requirements Management Policy and Responsibilities 7. Review Project Plan 8. Review by Senior Management 9. Show Commitment to the Plan	Approved Plan Acceptance Criteria

Exit Criteria: Agreement among the stakeholders on scope of task, assignments, and schedules. Resources and funds allocated. An approved plan.

Figure 6-2. Commitment Planning.

Elicitation

Participants: Customer, Project Management, Systems Engineering, Software Engineering, CM/QA, Functional Testing, Operational Testing, End User.

Entry Criteria: Agreement on the plan, program risks, resources and funding have been identified, responsibilities are allocated and understood.

Inputs	Processing	Outputs
Approved Plan Mission Need Change Derived Requirements	1. Conduct Fact-Finding 2. Capture Candidate Technical Requirements 3. Capture Non-Technical Requirements	Candidate Technical Requirements Non-Technical Requirements Supporting Rationale

Exit Criteria: An understanding of the candidate technical requirements has been obtained and supporting rationale documented.

Figure 6-3. Elicitation.

Analysis

Participants: Project Management, Systems Engineering, Software Engineering.
Entry Criteria: An understanding of the candidate technical requirements and supporting rationale.

Inputs	Processing	Outputs
Candidate Technical Requirements Non-Technical Requirements Supporting Rationale	1. Categorize and Prioritize 2. Determine Quality Attributes 3. Establish Traceability 4. Reconcile Requirements 5. Capture Decision Rationale	Information Requirements Initial Requirements Traceability

Exit Criteria: Informal requirements have been produced through assessment of the candidate requirements, requirements have been categorized, traceability established, and decision rationale has been captured and reconciled.

Figure 6-4. Analysis.

Formalization

Participants: Systems Engineering, Software Engineering, CM/QA.
Entry Criteria: Informal requirements produced through the assessment of the candidate requirements, categorized and prioritized data based requirements, establishment of traceability, and reconciliation of decision rationale.

Inputs	Processing	Outputs
Informal Requirements Requirements Traceability	1. Transform Requirements to Formalized Engineering Artifacts 2. Formalize Traceability of Engineering Artifacts 3. Place Formalized Engineering Artifacts under Configuration Control 4. Place Formalized Engineering Artifacts under Configuration Control	Formalized Requirements as engineering artifacts and traceability work products under CM

Exit Criteria: The formal requirements/formal engineering artifacts and formal traceability work products are placed under CM control.

Figure 6-5. Formalization.

Verification

Participants: Customer, Project Management, Systems Engineering, Software Engineering, CM/QA, Functional Testing, Operational Testing, End User.
Entry Criteria: Requirements as formal engineering artifacts and formal traceability work products under CM.

Inputs	Processing	Outputs
Formalized Requirements Formalized Engineering Artifacts Formalized Traceability Work Products	1. Perform Assessment 2. Verify Traceability 3. Resolve Deficiencies 4. Facilitate Agreement 5. Establish a Requirements Baseline	Baselined Requirements Verification Report Anomaly Report

Exit Criteria: Formalized engineering artifacts have been reviewed, approved and placed under CM control. Any deficiency reports generated during the review cause a return to a previous activity for investigation.

Figure 6-6. Verification.

Commitment/Acceptance

Participants: Senior Management, Customer, Project Management.
Entry Criteria: Baselined requirements and verification report.

Inputs	Processing	Outputs
Baselined Requirements Verification Report	1. Present Evidence of Baselined Requirements 2. Present Project Status 3. Formalize Commitment by All Stakeholders to Baselined Requirements 4. Obtain Approval to Proceed to the Next Process 5. Verification 6. Commitment/Acceptance	Authenticated Requirements Approval to Proceed

Exit Criteria: The customer has approved the baselined requirements as meeting the acceptance criteria. Approval to proceed to the next evolution of the requirements management process is obtained.

Figure 6-7. Commitment Acceptance.

7. APPENDIX C - SAMPLES AND TAILORING GUIDANCE

A number of samples of project implementations of the RMPM are provided herein. Each of the samples has been tailored to suit their unique requirements needs.

7.1 DOD-STD-2167A Requirements Management

The paragraphs below provide an example of the flow down of requirements in DOD-STD-2167A documentation, as well as an example of how the spiral model can be applied to a development effort that is using DOD-STD-2167A.

7.1.1 DOD-STD-2167A Document Hierarchy

Most programs follow a given software development standard such as DOD-STD-2167A, DOD-STD-1679A, or DOD-STD-498 that contain Data Item Descriptions (DIDs) for system documentation developed during the system life cycle. Several documents under these standards are considered key to the requirements process and form the basis for a requirements traceability analysis.

For a DOD-STD- 2167A program, this document set consists of the System/Segment Specification (SSS), System/Segment Design Document (SSDD), Interface Requirements Specification (IRS), SRS, Interface Design Document (IDD) and Software Design Document (SDD). These documents have a distinct parent-to-child relationship (called direct relationship) because starting with the SSS, the requirements flow down to each successive document and expand in number as a greater understanding of the system requirements is achieved. Figure 7.1.1-1 shows the hierarchical relationship between the DOD-STD-2167A requirements documents.

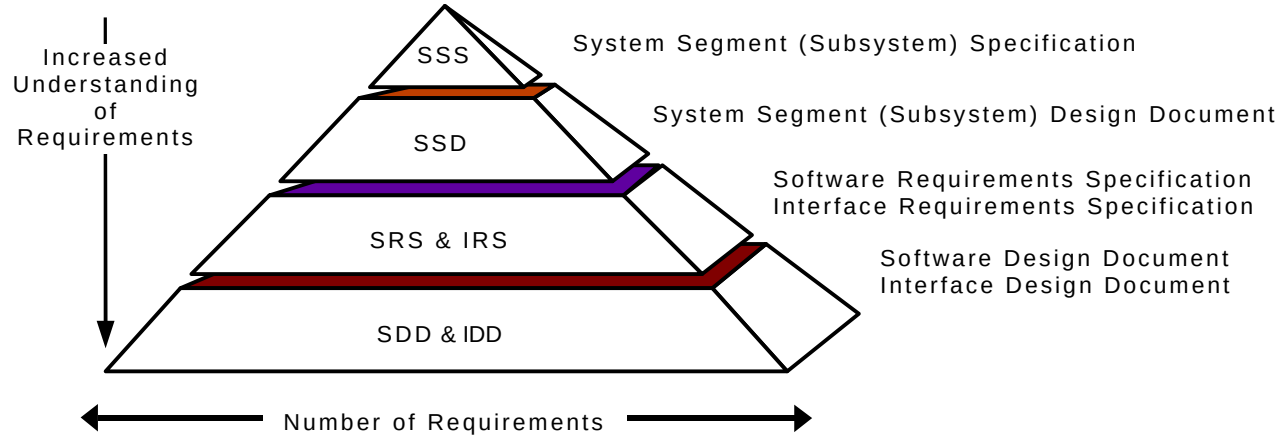


Figure 7.1.1-1. Requirements Tracing for DOD-STD-2167A Documents.

Within the system life cycle, the DOD-STD-2167A documentation represents only part of the requirements documentation that is covered under a traceability analysis. Other key documents that are traced or linked to the DOD-STD-2167A requirements documents are listed below.

- Mission Needs Statement (MNS)
- Operational Requirements Document (ORD)
- Cost and Operational Effectiveness Analysis (COEA)
- Test and Evaluation Master Plan (TEMP)

These documents, which are a combination of requirements documents and stand-alone documents, are cited in DODI 5000.2 and are used to support milestone decisions. Unlike the DOD-STD-2167A requirements documents, all the DODI 5000.2 documents do not have a direct hierarchical relationship where the requirements of a higher-level document flow down to a lower-level document. Instead, stand alone documents are developed that often contain additional requirements or clarify existing requirements but were not written with the intent of being a requirements document.

Requirements addressed in stand alone documents do not have a direct relationship with those requirements contained in a requirements document. Instead, they have a peer-to-peer or indirect relationship. This means that requirements must be consistent or non-conflicting. It also means that similar to a direct relationship, any requirements changes will impact the overall system requirements. For example, the COEA is not an expansion of the requirements in the ORD, yet the constraints, performance objectives, and measures of effectiveness stated in the COEA should be consistent with the ORD. This indirect linkage between the two documents is established and tracked during a traceability analysis.

Many programs also prepare a system specification based on MIL-STD-490B and include it as part of a contract solicitation. This specification, known as the A-Spec, is used by the selected contractor as the basis for the contractor-developed SSS. As shown in Figure 7.1.1-2, requirements flow down from the ORD, to the A-Spec, and then to the SSS reflecting the direct relationship that exists between these documents. In addition, the contract Statement of Work (SOW) often contains requirements that indirectly impact the A-Spec, SSS, or SRS and is usually included as part of a traceability analysis. Figure 7.1.1-2 summarizes the relationship among the key documents that typically make up the requirements traceability analysis for a DOD-STD-2167A program.

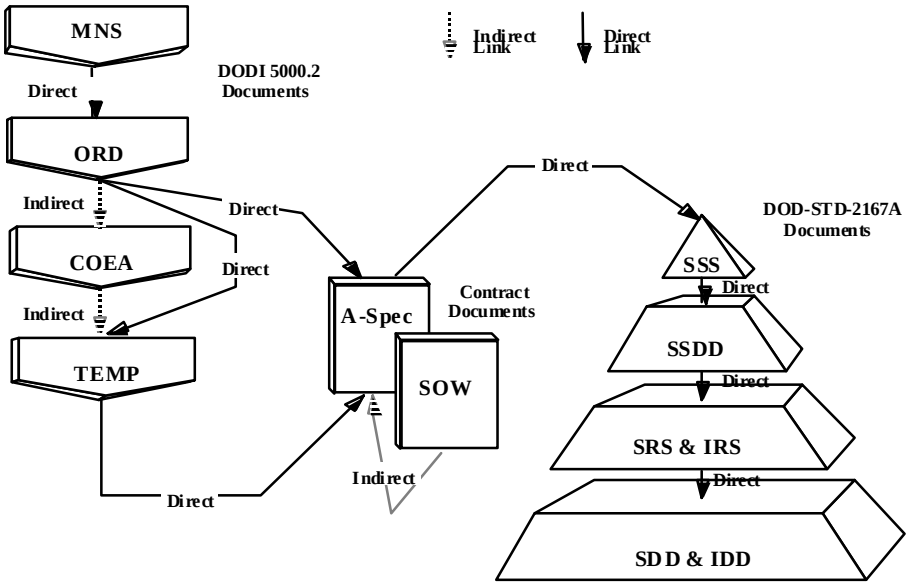


Figure 7.1.1-2. Tracing Requirements Among Requirements Documentation for a DOD-STD-2167A Program.

7.1.2 DOD-STD-2167A Example

The examples below illustrate how projects using the DOD-STD-2167A software development standard can manage requirements using the requirement management process model.

- Example 1 - A small project with minimal interfaces.
- Example 2 - A complex project with multiple CSCI/Hardware Configuration Item (HWCI) interfaces.
- Example 3 - Corrective action that requires changes to source documentation.

Example 1

Requirements management for a small system can be modeled using a single spiral. Each cycle of the spiral corresponds to a DOD-STD-2167A document. The example illustrated in Figure 7.1.2-1 shows one SSS, one SSDD, one SRS, and one SDD.

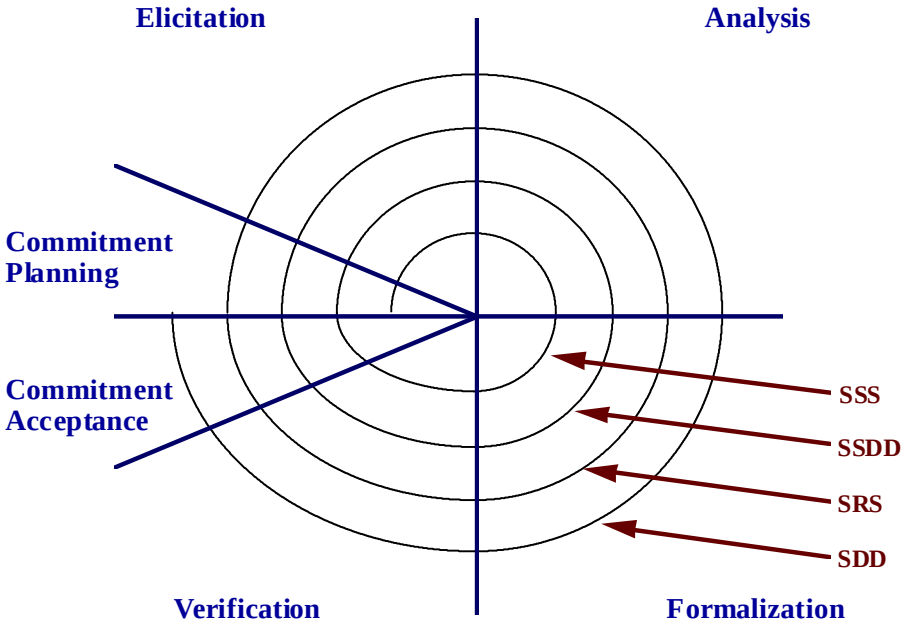


Figure 7.1.2-1. Single Spiral Model.

Example 2

The requirements management process for more complex systems can be modeled using subspirals. Subspirals can be spawned when a management or design decision is made that causes requirements documented in a single-source document to be allocated to multiple lower-level requirements documents for elaboration (e.g., allocating system requirements to multiple subsystems or to hardware and software). Figure 7.1.2-2 shows an example of a system where one SSS and one SSDD provide a complete description of the system requirements. These system requirements are allocated to multiple CSCIs and HWCIs, each of which has its own requirements management process represented by a subspiral.

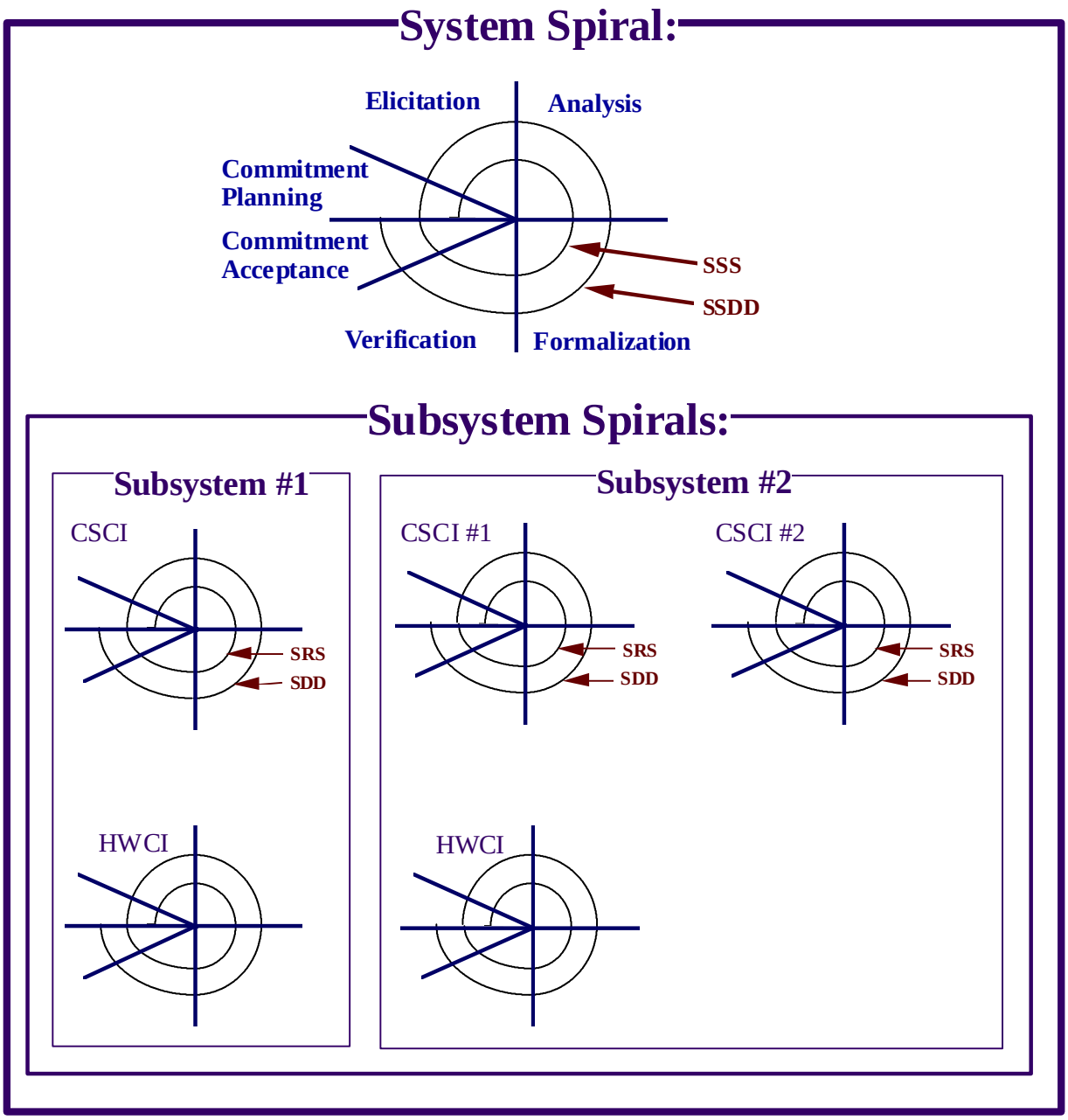


Figure 7.1.2-2. Multiple Spiral Model.

Example 3

Inevitably, software developers and systems engineers working on requirements will determine that a source requirement is incorrect or requires elaboration. At this point, the corrective action process is initiated and a deficiency report is generated. Investigation of the deficiency report will indicate the highest level source requirements document which requires modification. Once the highest level source document is identified, a subspiral is spawned, and the requirements management process is repeated to ensure that the requirements change is propagated to all lower-level documents. Figure 7.1.2-3 shows a spiral for a simple DOD-STD-2167A project with a subspiral showing the requirements management process for correcting a defect identified in the SSDD and SRS while working on the SDD.

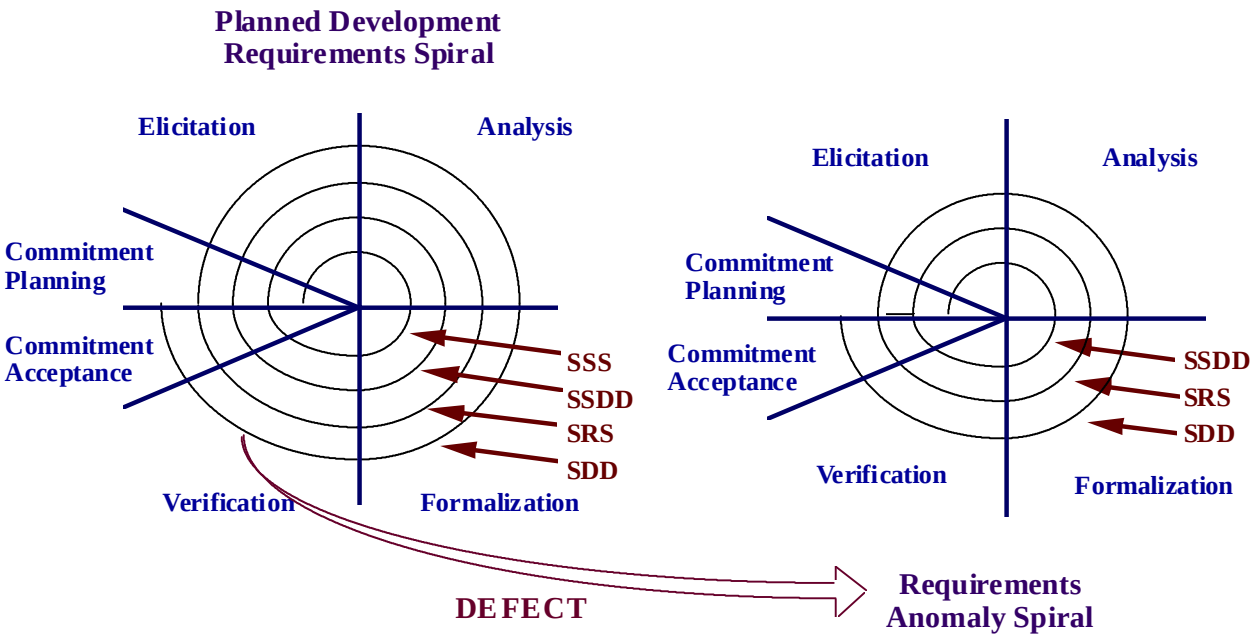


Figure 7.1.2-3. Corrective Action Spiral Model.

7.2 F/A-18 Requirements Flow Example

The F/A-18 program began prior to the implementation of DODI 5000.2 and DOD-STD-2167A; therefore, the flow of requirements from MNS through the detailed system software design has been tailored to meet the needs of the program while also complying with the applicable instructions. In addition, most of the milestone decisions identified in DODI 5000.2 do not apply to the software activities as the aircraft is in a production support phase (i.e., post milestone III).

The F/A-18 system software is made up of over 60 different CSCIs in over 30 different subsystems. Software releases are scheduled every two to three years. Each software release is organized into a system or "block" update. The update schedules of all the subsystems are coordinated to be released as a system package. Each update includes a combination of enhanced, adaptive and corrective maintenance items. In addition, there are generally a few new subsystems.

The F/A-18 program utilizes a "Functional Requirements Document" (FRD) to identify all system changes for each block update. The FRD acts as the single reference point for the system requirements after the ORD. The FRD contains a section for each major requirement (e.g., Incorporate Global Positioning System (GPS) Navigation Capabilities). For each of the major requirements, the FRD contains the information listed below.

- Statement of Requirement (SOR)
- Operational intent
- Statement of Functionality (SOF)

The SOR is a simple statement of the system-level requirement for this particular need (e.g., provide time-to-impact for Laser-Guided Bombs (LGB)).

The operational intent is a statement of how and why the requirement is necessary (e.g., provide the time to impact for all LGBs to direct the laser designators when to illuminate the target). This assists all stakeholders, end users and customers in attaining and maintaining a mutual understanding of what may be required to meet the requirement.

The SOF is a detailed, system-level description of how this requirement is going to be mechanized in the aircraft. This section of the FRD is normally not completed until after software preliminary design. Much of what goes into this section comes from the SSDD.

The most unique and interesting aspect of the FRD is how it is utilized as a critical part of test and evaluation. OPNAVINST 5000.42D *Test and Evaluation Policy and Procedures* requires that the program manager for each project forward a "statement of functionality" to COMOPTEVFOR for use in the Software Qualification TEMP. The F/A-18 program provides the FRD for use in this capacity. This ensures that the system software development is developed against the same set of requirements as the operational test personnel will be testing. Figure 7.2-1 illustrates the relationship between the documents described in this paragraph.

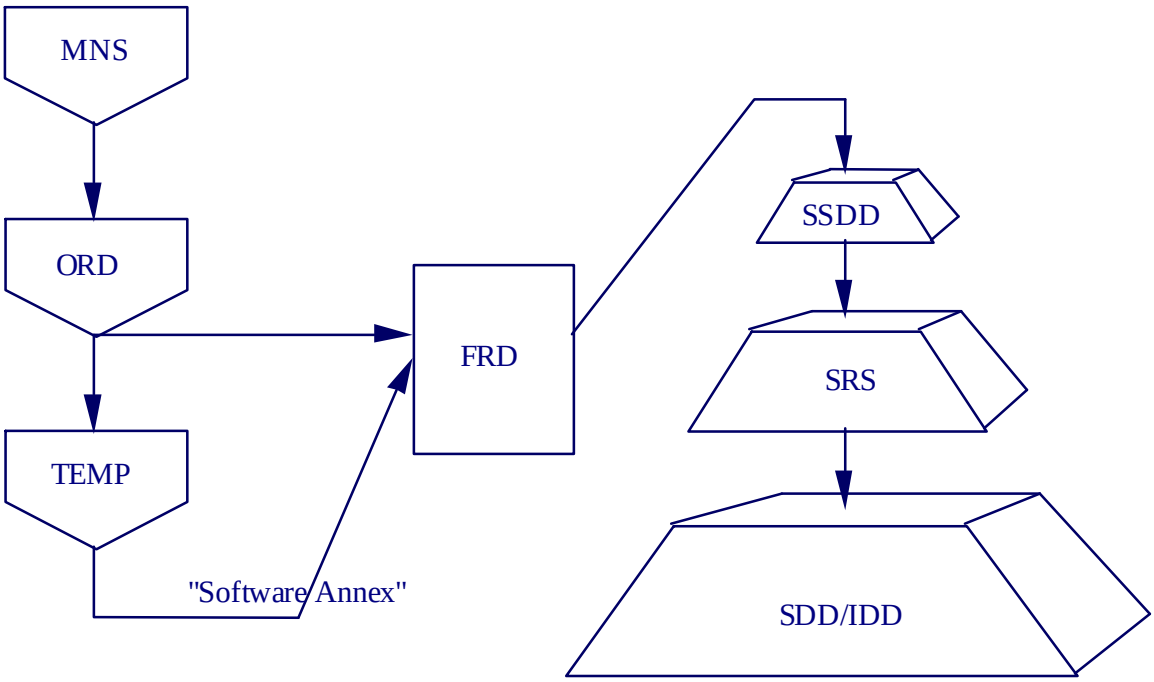


Figure 7.2-1. F/A-18 Document Hierarchy.

7.3 Project Example Using the Requirements and Traceability Management (RTM) Tool

The paragraphs below describe the requirements management method implemented by the Automated Communication Management System (ACMS) project using the RTM Tool. This example is not intended to endorse any specific tool but rather to provide a sample of one project's tool and application of that tool in performing requirements management in a realistic situation. This information was taken from appendix C of the draft version of the *Software Development Plan (SDP) for the ACMS Project*..

7.3.1 Requirements Engineering

The objective of Requirements Engineering is to take the customer's specific, written requirements and translate them into a clear and unambiguous set of derived requirements to control the design, development, implementation and testing of the ACMS final system so that we deliver a product that meets customer expectations. We are using the RTM tool to accomplish this objective, because of its extensive requirements engineering capability and its ability to maintain requirements traceability back to the original customer (source) documents.

RTM (Version 2.3.3) is a Requirements Engineering CASE Tool running on a Sun workstation under a Solaris 2.4 operating system with Oracle 7.0.16 as its relational database. RTM complies with a number of open standards (Portable Operating System Interface for Compute Environments (POSIX) for portability, Structured Query Language (SQL) for database access, and X-windows for GUI), and interfaces with other software tools (such as Software through Pictures (StP)) to provide:

- requirements identification and engineering
- requirements traceability throughout the project life cycle
- construction and maintenance of other system entities
- audit trails
- compliancy checking
- structured and/or object-oriented analysis and design
- configuration management
- documentation support

It is expected that all ACMS project teams at some point in the development process will use RTM. Hence, every person on the project is theoretically eligible to use RTM. Since RTM is network compatible, all ACMS project personnel will be granted read access privileges to RTM data, whereby they can perform activities such as display the diagrams, perform attribute and keyword sorts, manipulate and/or trace data and print reports. However, RTM is set up to restrict, write, create and delete access to specific project personnel. The RTM system administrators in accordance with Project Management direction assign the accesses.

7.3.2 User Support Concept

In general, the project will produce a requirements traceability matrix for each major project document (such as SRSs, IRSs and SDDs) to ensure that each phase of the project conforms to all requirements. As each matrix is produced, the project will be able to determine whether all requirements for that phase have been met, whether some requirements were not met, or whether unsubstantiated requirements were generated.

The Requirements Management Team (RM Team) is responsible for overseeing RTM usage, for providing informal training as required and for establishing the overall schema for the requirements database. Further, the RM Team alone will input source requirements, maintain the "master file" of all ACMS requirements and produce requirements traceability matrices.

This paragraph will discuss requirements database "baselining" and how it might change access privileges.

7.3.3 Implementation

RTM is designed to "engineer" the customer's original (source) requirements so that the derived requirements can be used by software developers, system engineers, testers, IV&V and other ACMS project personnel. RTM supports these project personnel by associating each source requirement with ACMS project related characteristics such as requirement class, attributes and key words. Once this is done, project personnel can then manipulate the RTM database to support their specific need (e.g., design, implementation, testing or verification).

7.3.3.1 Source Documents

Source documents are defined as any documents that contain ACMS software requirements. Source documents may contain a complete list of all requirements for the entire system, a section of the system, or a partial list of system requirements. Consequently, some documents may contain requirements that are duplicated in other source documents. RTM facilitates extracting (stripping) requirements from source documents, automatically assigns a project-unique identifier to all requirements and allows source documents to be reconstructed from engineered requirements. RTM can also focus multiple, similar requirements derived from different source documents into one requirement while maintaining traceability back to the original source document. The ACMS RM Team enters requirements from source documents into RTM.

Source documents for ACMS include:

ACMS Segment Specification (A-Spec)	This document incorporates Mission Planning Element (MPE), FRD, HMI, security and open system requirements
Interface Control Documents (ICDs)	These documents specify the requirements interfaces between ACMS and other entities.

7.3.3.2 Class Definitions

RTM uses three general types of data classes:

Requirement Class	Defined by RTM project administrator. Used group customer defined requirements data.
RTM Class	Used to group ACMS project generated data within RTM.
CASE Tool Class	Used to group ACMS project generated CASE Tool data such as StP.

Individual requirement classes are linked to other classes by a relationship. These relationships are defined by the user and are the primary means of requirements traceability. Each class can be linked to as many other classes as desired. The link can be established as a 1-to-1 (direct link), 1-to-many (requirement expansion), or many-to-1 (requirement focusing) relationship.

For the ACMS project, source documents are defined as being of class “requirement.” When a source document is “stripped” of its requirements, a relationship, or link (which RTM depicts as a rectangle) is established between the source document and class ACMS -- the central repository requirements class for the project. Class ACMS collects ALL requirements for the project. The relationship between source documents and class ACMS is shown in Figure 7.3.3.2-1.

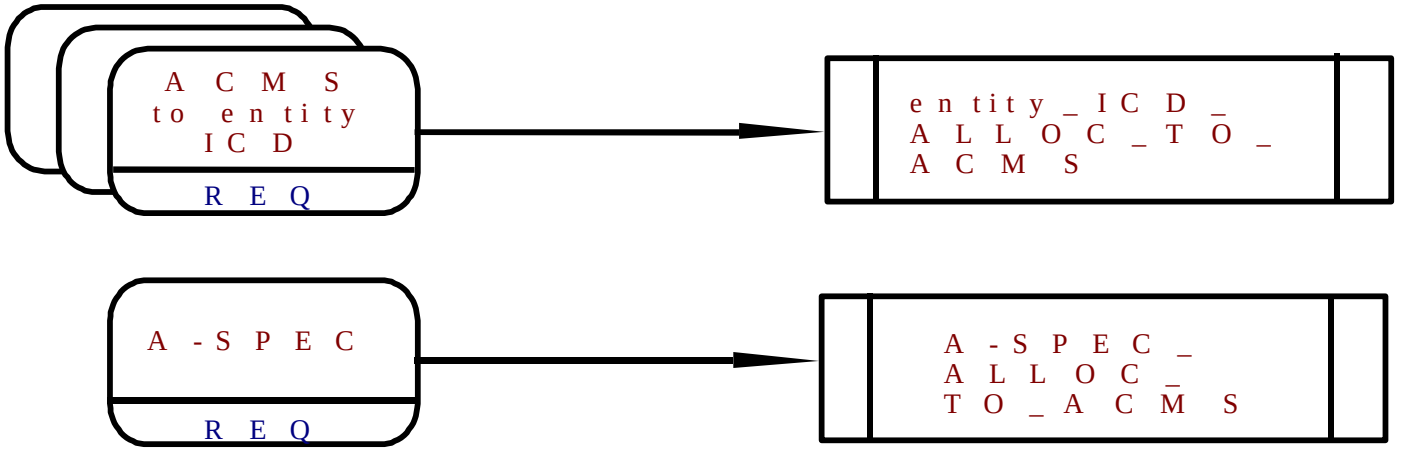


Figure 7.3.3.2-1. Relationship of Source Documents to Class ACMS.

7.3.3.3 Attributes

In addition to tracking requirements, RTM allows the user to produce listings of similar/related requirements through the use of attributes. Each requirement has attached a list of attributes that can be used to manipulate, sort and view lists of requirements that have a specific value in attribute fields. Each class has its own list of attributes. However, all individual requirements assigned to a single class have the same attribute lists.

RTM has two types of attributes, user specified and RTM specified. The user can choose from one of six different user defined attribute types.

Alpha-numeric	1-75 alpha-numeric characters.
Numeric	1-75 numbers.
Date	dd-mmm-yyyy or mmm-dd-yyyy.
Configurable	User can set up a finite list of allowed entries (pick-list). One or more selections can be made.
Free Text	Paragraph type entry of up to 16K characters.
File	A pointer to a specific computer file.

The RTM defined attributes are automatically allocated and assigned by the RTM software. They are:

Requirement Key	A project-unique identifier
Requirement status	Whether the requirement is current, replaced by a child or focused into another requirement
Source requirement	The ultimate "ancestor" requirement if this is a child
Original editor	The name of the individual who performed the requirements stripping (determined at logon)
Document ID	The name of the source document
Paragraph ID	The source document paragraph number from which the requirement was stripped
Time/Date created	Self-explanatory
Last modified	Self-explanatory

7.3.3.4 Keywords

In addition to tracing individual requirements through class definitions and associated links, RTM allows requirements to be related (grouped) by assigned keywords. ACMS project-specific keywords are assigned to each requirement to facilitate sorting/manipulating related requirements. Any one requirement can have many keywords assigned or can have no keywords assigned.

To help project personnel with such groupings, the RM Team has associated as many keywords with a given requirement as possible. Consequently, it is rare for an ACMS requirement to have no, or only one, keyword. The RM Team maintains an off-line listing of all RTM keywords. This listing is available to project personnel.

7.3.3.5 Requirements Normalization

There will be many instances in the ACMS project where a stated "requirement" is actually a set of requirements within a single statement. If left as stated, these multiple requirements could create misunderstandings as to the customer's intent. Thus, it becomes important to restate (or "normalize") a requirement so as to eliminate these potential misunderstandings. Requirements normalization, then, is the process whereby ambiguous or unclear requirements are restated as discrete entities so as to remove any ambiguity. This process is key because it also:

- breaks requirements statements into discrete requirements
- checks for testability
- checks for closure clarity
- identifies the source point for the requirement

The RM Team will normalize requirements after they have been linked into the ACMS database. The requirements paragraphs, as stated in the source documents, will be broken up into individual, discrete, testable requirements. The normalization process will likely produce a large number of individual requirements from the original source requirements. It should be noted, however, that normalization has no effect on traceability. Rather, it establishes a set of children/grandchildren that clarify/quantify the parent requirement.

7.3.4 Schema

One of the more important features of RTM is the user's being able to define a project-specific schema. The ACMS project is organized around a document hierarchy, beginning with source documents

provided to the project (e.g., the A-Spec, ICDs, IDDs) and continuing through project generated documentation: SSDD, SRS, IRS, SDD, etc. The RTM tool schema parallels this document hierarchy. For the ACMS project, individual source documents will have their requirements entered into separate classes. Each of these classes will then be linked to a single class that will group all of the ACMS requirements. As this linking is done, similar or identical requirements from different documents will be focused into one requirement in the ACMS class.

Just as the project has an SSDD that captures all the requirements of the A-Spec, the ICDs, the IDDs, etc., RTM has a comparable Class SSDD (of type RTM) that captures all of the class ACMS requirements that apply to the SSDD. Further, the tool classes (of type RTM) that correspond to the test requirements that the software Test Team defines to test CSCIs, CSCs, CSUs, etc.

The Software Test Plan will be entered into RTM just as any source document. Its requirements will then be linked to the Test Plan SRS class then further linked to the specific CSC. In this manner, test traceability will be maintained. Reports can be produced that provide a list of test requirement traceability from a specific test to a CSC and from a CSC back to the test.

Figure 7.3.4-1 displays the RTM schema. (NOTE: in the interest of simplicity, the relationship boxes have been deleted from the figure and arrows show relationships.)

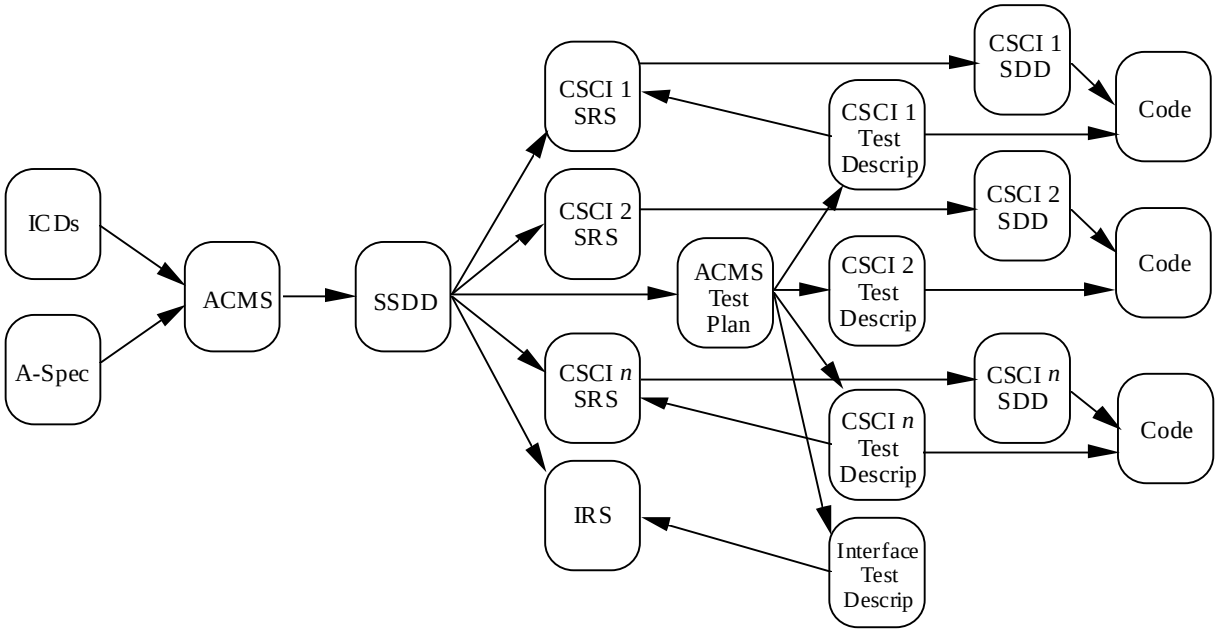


Figure 7.3.4-1. RTM Tool Schema.

7.3.5 External Interfaces

7.3.5.1 Software through Pictures (StP)

The ACMS project will use StP as its software design CASE tool. RTM provides an interface to capture and track StP design information. This interface will see that all requirements are addressed in the design phase and ensuring that all design objects are linked to a requirement. Once this interface is established, it will be possible to view, link and trace all StP objects just like other RTM classes.

7.3.5.2 FrameMaker

FrameMaker will be used for document and report production. RTM allows for the direct interfacing with FrameMaker to easily accomplish this task.

7.3.5.3 Team Interfaces

This paragraph will discuss the RTM interfaces with the following ACMS project teams:

- Software Development Team
- Systems Engineering Team
- Software Test Team
- IV&V Team
- Software Quality Assurance (SQA) Team

7.3.6 Document Support Concept

The ACMS project is a document-driven project, in that requirements are stated in formal specification and are then redefined in various project documentation. RTM will contain both the requirements from the formal specifications, as well as the project derived requirements against which each step of the development process must be measured. Thus, RTM will participate in each step of the development process.

RTM can produce a variety of project documentation. It can produce informal reports/listings or, using FrameMaker, it can produce formal reports or project documents. In addition, it can recreate the source documents using the requirements that have been normalized and engineered during the requirement engineering process. Figure 7.3.6-1 illustrates this activity.

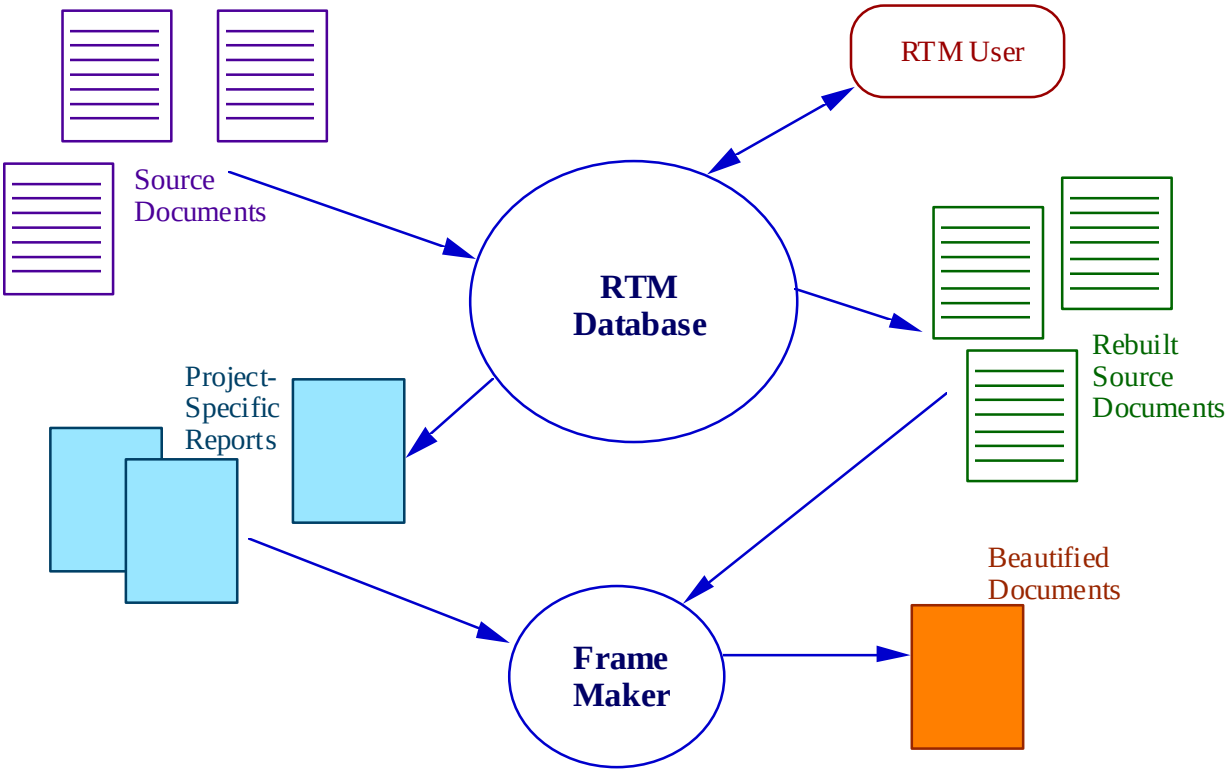


Figure 7.3.6-1. RTM Document Support within the ACMS Project.

7.3.6.1 Reports

As noted above, RTM provides for a wide variety of reports. These reports will be customized to meet the needs of the project. The reports listed below have been identified for project usage (NOTE: this is not a complete listing as more reports can be developed as the project progresses or as the need arises.):

Requirements Traceability Matrix	Compares requirements for each project phase against applicable project documents to ensure requirements have been met
Discrepancy Listing	Notes instances where project requirements have not been met

7.3.6.2 Discrepancy Tracking

This paragraph will address how discrepancies will be recorded and monitored.

8. APPENDIX D - REQUIREMENTS MANAGEMENT PROCESS CHECKLIST EXAMPLE

A process checklist provides a mechanism for verifying that a process is being followed and identifies areas where a process modification can result in process improvement. The checklist in Figure 8-1 provides an example of a requirements management process checklist that can be tailored to meet project-specific objectives. This example was developed based on the requirements management process contained in Section 4.

Requirements Management Process Checklist	Criteria Satisfied	
	Yes	No
Commitment Planning: Were all stakeholders identified?		
Was the acceptance criteria for this spiral cycle defined?		
Were non-technical requirements identified and documented?		
Has the project plan been developed or updated?		
Has the project's risk been assessed?		
Has the requirements management policy been reviewed?		
Have the metric collection points and schedule for requirements management been identified?		
Has the project plan been reviewed?		
Did senior management review the project plan?		
Was commitment to the project plan obtained?		
Elicitation: Was information concerning the problem's domain, open issues and resolution identified?		
Were the candidate technical requirements captured?		
Were non-technical requirements captured?		
Analysis: Were the requirements decomposed?		
Were the quality attributes for each requirement determined?		
Was traceability of the requirements established?		
Was a reconciliation of the requirements performed?		
Was the rationale for any decisions captured?		
Were modifications to the requirements reflected in the project plan?		

Figure 8-1. Sample Requirements Management Process Checklist (Page 1 of 2).

Requirements Management Process Checklist	Criteria Satisfied	
	Yes	**No**
Formalization: Were the informal requirements and supporting information documented?		
Were formalized work products developed?		
Were formalized work products placed under CM?		
Verification: Were the formalized requirements inspected for the quality attributes?		
Were inconsistencies among the requirements identified and corrected?		
Were redundant requirements identified and corrected?		
Were deficiency reports generated and placed under CM?		
Was the project plan updated to reflect changes made as a result of deficiency report resolution?		
Are the formalized requirements traceable?		
Have the stakeholders examined the formalized engineering artifacts and verified that they represent customer and end user requirements?		
Were the formalized engineering artifacts placed under CM?		
Commitment Acceptance: Were requirements metrics presented to the customer?		
Was the project status presented to the customer?		
Did the stakeholders approve the baselined requirements?		
Did the customer provide approval to proceed?		
Were periodic or event driven reviews of the requirements held with project management?**		
Did QA audit the activities and products of requirements management?**		

** *Open for discussion*

Figure 8-1. Sample Requirements Management Process Checklist (Page 2 of 2).

9. APPENDIX E - SUGGESTED METRIC COLLECTIONS

The CRG has developed the *Metrics Handbook* to address metric collection and analysis for development projects. The topics below are additional metric points that apply to requirements management.

Defect. One of the main goals of RM is cost avoidance through identification of requirements defects before they progress to design defects and code defects (see B. Boehm, Software Engineering Economics, 1981). Tracking the number of each of these types of defects has utility in several ways. First, it aids in cost accounting and estimation, as cost-to-fix can be allocated to each defect. Second, it facilitates process improvement: numerous requirements defects will indicate a flawed requirements analysis process; consistent numbers of design and code defects will indicate a weakness in detecting and correcting requirements defects.

Attribute oriented. Included among the RM activities specified in this handbook is the task of reviewing each requirement against a set of project-defined quality attributes. Requirements that do not reflect the necessary attributes require further refinement (e.g., investigation, communication with customer). Tracking the degree to which each requirement satisfies the set of quality attributes will aid in determining overall quality of the requirements set. This metric can be tied to both the Defect and Staff Hour metrics. A requirement that does not satisfy the set of quality attributes may be categorized as defective and referred to an adjudicator. Effort to correct these defective requirements should also be tracked for cost accounting/estimation and process improvement purposes. In addition to quality attributes, many other attributes can be assigned to requirements. It is then possible to sort, view, and enumerate lists of requirements that have a specific value in attribute fields.

Staff hours spent on requirements management. The best way to refine future estimates for specific RM activities is to collect data on each RM activity as it occurs. In addition to its value for estimation, staffhour data also is necessary for evaluating process improvement initiatives. Comparing the number of hours to complete activities using different procedures allows an objective means for determining the success or failure of new or modified procedures. Finally, staffhour metrics on on-going activities can help to identify possible overruns and identify contingency plans early.

9.1 Product Measurements

1. Number of requirements within a formalized engineering artifact. (can be subdivided by requirement type)
2. Number of deficiencies attributed to a formalized engineering artifact.
3. Number of formalized requirements.
4. Number of requirements by state as related to their ability to meet the quality attributes (i.e. TBD, incomplete, unstable as to their specification and detail. ??? etc.)

9.2 Process Measurements

1. Staff hours utilized to produce the:
   - Formalized requirements
   - Formalized engineering artifacts
   - Formalized traceability work product

10. APPENDIX F - DEVELOPMENT MODELS

Since the late 1960s, several process models for software have been developed, each offering a unique perspective, insights and solutions to software life-cycle problems. A given process model generally constrains the sequence in which work is performed starting when the product is conceptualized and ending when the product has satisfied the acceptance criteria.

Three common software development models (waterfall, incremental development, and spiral) are summarized in this Appendix. The requirements management framework outlined in this Handbook can be applied to any of the three lifecycle models.

10.1 Waterfall Model

The waterfall model consists of a set of software development phases that are generally considered to be the conventional or "classical" software life cycle.

In the Requirements Analysis and Specification phase, the user's requirements are specified in a series of system and software requirements documents. The Architectural Design phase consists of structuring the requirements into constituent components. Detailed Design consists of developing the algorithmic details for each system component. During Coding, each component is coded in a programming language selected for the project.

A problem with the waterfall model is that it is primarily a linear process that tends to restrict the backward movement required for corrective action. Figure 10.1-1 shows the classic waterfall.

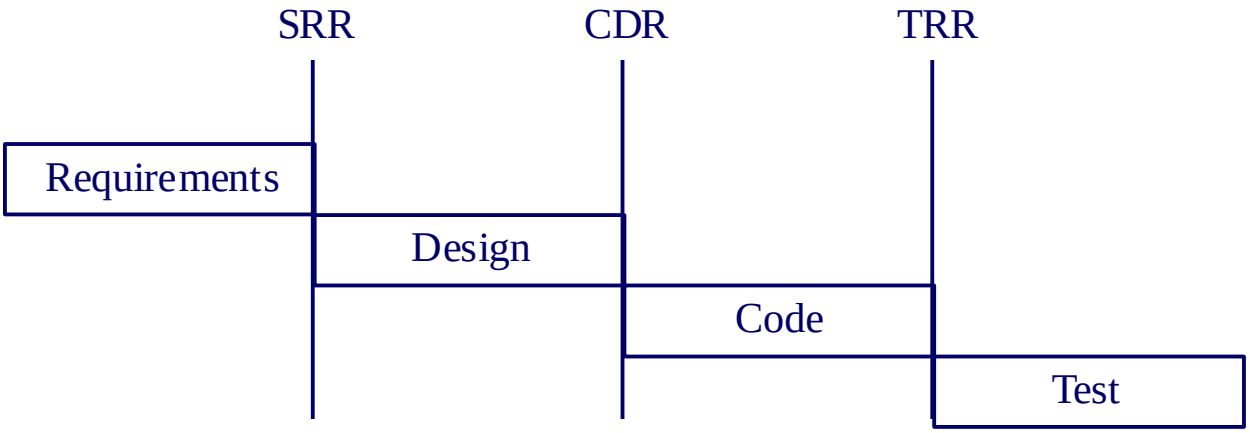


Figure 10.1-1. Classic Waterfall.

10.2 Spiral Model

From [Gomaa 1993]:

"The spiral model provides an approach for integrating prototyping and incremental development with the waterfall model. The Spiral Model is an iterative life cycle in which each loop of the spiral represents one iteration. The radial coordinate represents cumulative cost. An important aspect of each iteration is a risk assessment of the project. The areas of greatest uncertainty and potentially significant problems are identified. Prototypes are developed and evaluated to help alleviate the areas of greatest risk. For example, if user requirements are not clearly understood, a throwaway prototype is developed. During design, a prototype could be developed to assess areas of greatest performance risks. Implementation only begins when the risks have been substantially reduced to a level considered acceptable by management."

The spiral model exhibits the following characteristics:

- Is intentionally not tied to a specific level of requirements maturity or abstraction.
- Can be used to define and evolve requirements that will best meet the project-unique objectives and constraints.
- Will help define an incremental process to generate complete, correct, detailed and unambiguous requirements as necessary to support system acquisition or development.

The spiral model was originally developed by Boehm (1986, 1988) to address the known problems in more conventional (primarily waterfall) process models.

A commitment gate and four main functional activities represent the spiral model. The activities include:

- Determining objectives, alternatives and constraints
- Evaluating alternatives
- Identifying and resolving risks
- Developing and verifying the product
- Planning the next phase

A cycle traverse all of the major functions, represented by a 360 degree rotation in Figure 10.2-1. For each completed cycle, some aspect of the product has matured by a specified amount.

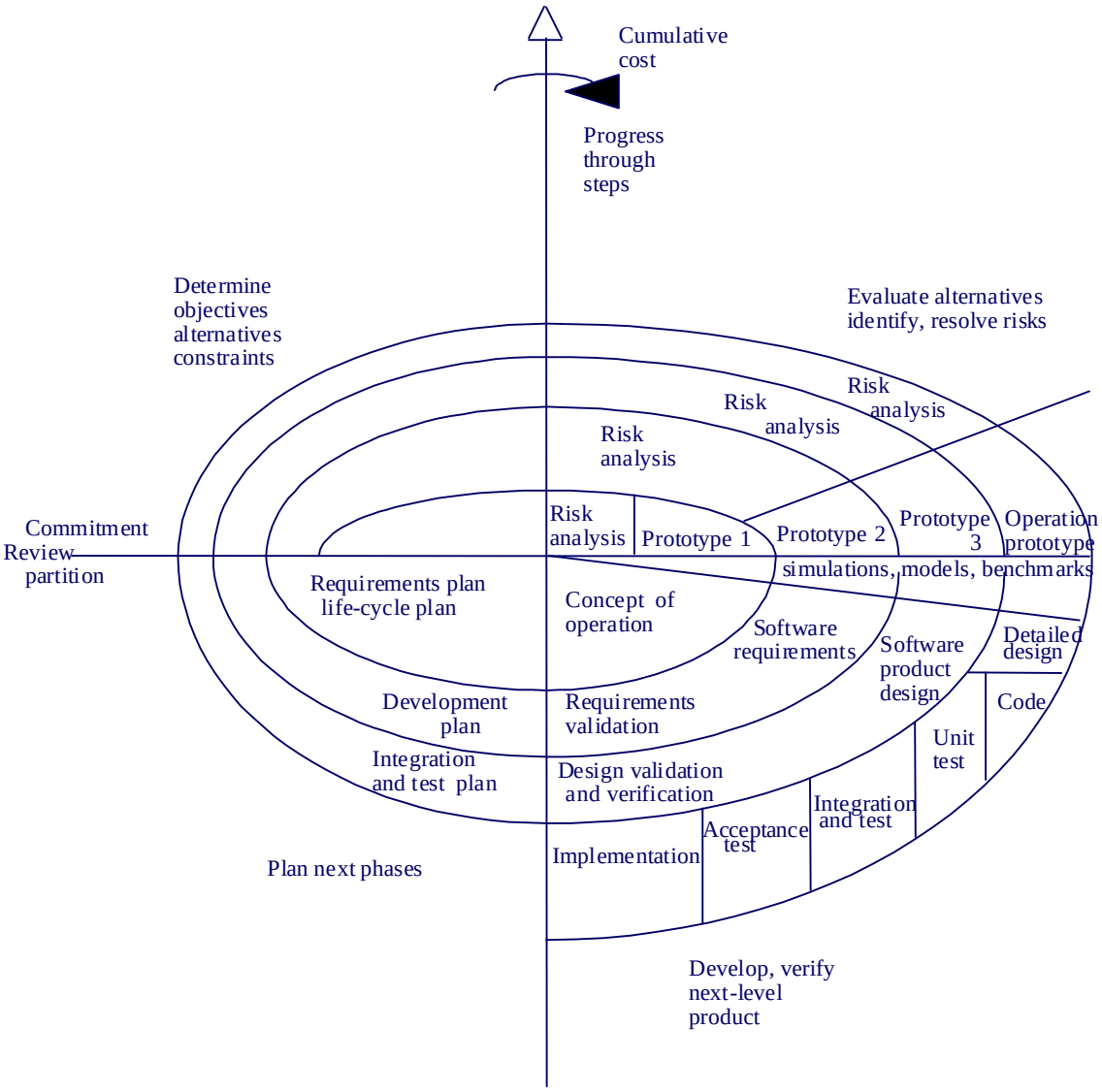


Figure 10.2-1. The Spiral Model.

For the requirements management process model, a cycle starts on the upper-left sector with the activities associated with **Commitment Planning**, then moves clockwise to requirements **Elicitation**, perform **Analysis**, develops a formalized representation of the requirements in **Formalization**, then to Verification of the evolving requirements baseline, and onto Commitment Acceptance. The transition from **Commitment Acceptance** to the next cycle is based upon the commitment to the requirements developed during the current cycle. Figure 10.2-2 shows the Requirements Management Process Model (RMPM).

Figure 10.2-2. The Requirements Management Process Model (RMPM).

Each cycle of RMPM is meant to be repeated using the knowledge gained and lessons learned from the previous cycles; that is, all sectors of the cycle are traversed one at a time. A cycle is a complete traversal of all activities that, when completed, matures the product by the amount defined in cycle-specific objectives. A spiral is one or more cycles that, when combined, accomplished a specific objective, such as an Operational Requirements Document (ORD), an A-Level Specification (A-Spec), one or more Software Requirements Specifications (SRSs), or even Software Design Documents (SDDs). A spiral may represent the complete cycle or may include only the activities necessary to meet one or more of the evolutionary states of requirements development.

By identifying and expressing a specific objective in the context of the conceptual Requirements Spiral Model (RSM) model, the process can be modified to meet project-specific objectives. In cases where the problem is well understood, it is possible to successfully front-end a waterfall, or top-down, approach that implements the requirements into design, code, integration and test of the program under acquisition or development.

10.2.1 Subspirals

The subspiral is generally a smaller version of the main spiral but follows the same model. Within each spiral, the cycles may be sequential or parallel, but no implied ordering is found among the cycle of different spirals. A subspiral is initiated to take corrective action and will define its own cycles as it addresses its objective. The information produced by a corrective action subspiral will feed the main spiral, most likely affecting decisions during **Verification** on the baseline requirements.

In addition, the strategy for parallel development of subproducts, such as SRSs for multiple Computer Software Configuration Items (CSCIs), can be defined and managed through the use of subspirals. During the **Commitment Planning** phase, parallel development of subproducts can be planned and initiated. Where there are several subspirals managing the parallel development of subproducts, a

project may choose to use the main spiral for subproduct management and integration activities, to develop the more important subproduct, or to develop the subproduct with the highest risk.

10.3 Iterative Build

From [Gomaa 1993]:

"The evolutionary prototyping approach is a form of incremental development, in which the prototype evolves through several intermediate operational systems into the delivered system. This approach can help in determining whether the system meets its performance goals for testing critical components of the design and for reducing development risk by spreading the implementation over a longer time frame."

"One objective of the evolutionary prototyping approach is to have a subset of the system working early that is then gradually built upon. It is advantageous if the first incremental version of the system tests a complete path through the system from external input to external output."

11. APPENDIX G - COMPLIANCE MATRIX

Appendix G shows the traceability of a external requirements to the information contained in the *Requirements Management Handbook*.

11.1 Capability Maturity Model (CMM) Requirements

CMM V1.1	*RM Guidebook Paragraph Number*	*Comments*
Requirements Management **Goals:** **G1.** System requirements allocated to software are controlled to establish a baseline for software engineering and management use.	Section 4	
G2. Software plans, products and activities are kept consistent with the system requirements allocated to software.	Section 4.1 Section 4.6	
Commitment to Perform: **C1.** The project follows a written organizational policy for managing the system requirements allocated to software. This policy typically specifies that:		
• The allocated requirements are documented.	Section 4.4	
The allocated requirements are reviewed by:		
• the software managers, and	Section 4.6	
• other affected groups	Section 4.5 Section 4.6	
• The software plans, work products and activities are changed to be consistent with changes to the allocated requirements.	Section 4.1 Section 4.6	
Ability to Perform: **A1.** For each project, responsibility is established for analyzing the system requirements and allocating them to hardware, software and other system components. This responsibility covers:		
• Managing and documenting the system requirements and their allocation throughout the project's life.	Section 4	

CMM V1.1	*RM Guidebook Paragraph Number*	*Comments*
• Effecting changes to the system requirements and their allocation.	Section 3 Section 4	Activity Interactions
A2. The allocated requirements are documented. The allocated requirements include:		
• The non-technical requirements (i.e., the agreements, conditions and/or contractual terms) that affect and determine the activities of the software project.	Section 4.2	
• The technical requirements for the software.	Section 4	
• The acceptance criteria that will be used to validate that the software products satisfy the allocated requirements.	Section 4.1	
A3. Adequate resources and funding are provided for managing the allocated requirements.		
• Individuals who have experience and expertise in the application domain and in software engineering are assigned to manage the allocated requirements.	Section 4.1	
• Tools to support the activities for managing requirements are made available.	Section 4.1	
A4. Members of the Software Engineering Group (SEG) and other software-related groups are trained to perform their requirements management activities,	Section 4.1	
Activities Performed: **A1.** The SEG reviews the allocated requirements before they are incorporated into the software project.		
• Incomplete and missing allocated requirements are identified.	Section 4.5	
• The allocated requirements are reviewed to determine whether they are:		
• Feasible and appropriate to implement in software.	Section 4.5	
• Clearly and properly stated.	Section 4.5	
• Consistent with each other.	Section 4.5	

CMM V1.1	*RM Guidebook Paragraph Number*	*Comments*
• Testable.	Section 4.5	
• Any allocated requirement identified as having potential problems is reviewed with the group responsible for analyzing and allocating the system requirements, and necessary changes are made.	Section 4.5	
• Commitments resulting from the allocated requirements are negotiated with the affected groups.	Section 4.5 Section 4.6	
A2. The SEG uses the allocated requirements as the basis for software plans, work products, and activities. The allocated requirements are:		
• Managed and controlled.	Section 4.1	
• The basis for the SDP.	Section 4.1	
• The basis for developing the software requirements.	Section 4.2	
A3. Changes to the allocated requirements are reviewed and incorporated into the software project.		
• The impact to existing commitments is assessed, and changes are negotiated as appropriate.	Section 4.1	
• Changes to commitments made to individuals and groups external to the organization are reviewed with senior management.	Section 4.1 Section 4.6	
• Changes to commitments within the organization are negotiated with the affected groups.	Section 4.1 Section 4.6	
• Changes that need to be made to the software plans, work products, and activities resulting from changes to the allocated requirements are:	Section 4.1	
• Identified.	Section 4.1	
• Evaluated.	Section 4.1	
• Assessed for risk.	Section 4.1	
• Documented.	Section 4.1	
• Planned.	Section 4.1	
• Communicated to the affected groups and individuals.	Section 4.1	

CMM V1.1	*RM Guidebook Paragraph Number*	*Comments*
•Tracked to completion.	Section 4.1 Section 4.6	
Measurement and Analysis: **M1.** Measurements are made and used to determine the status of the activities for managing the allocated requirements.	Section 4.1	
Verifying Implementation: **V1.** The activities for managing the allocated requirements are reviewed with senior management on a periodic basis.	Section 4.6	
V2. The activities for managing the allocated requirements are reviewed with the project manager on both a periodic and event-driven basis.	Section 4.1 Section 4.6	
V3. The SQA group reviews and/or audits the activities and work products for managing the allocated requirements and reports the results. At a minimum, these reviews and/or audits verify that:	Section 4.5	
• The allocated requirements are reviewed and problems are resolved before the SEG commits to them.	Section 4.5	
• The software plans, work products and activities are appropriately revised when the allocated requirements change.	Section 4.1	
• Changes to commitments resulting from changes to the allocated requirements are negotiated with the affected groups.	Section 4.1	

11.2 MIL-STD-499B Requirements

MIL-STD-499B	*RM Guidebook Paragraph Number*	*Comments*
4.3 Requirements Analysis The purpose of this activity is to serve as a bridge between customer requirements and system specific requirements from which solutions can be generated.	Section 4.2 Section 4.3	
The contractor shall analyze customer needs, objectives and requirements in the context of customer missions, utilization environments, and identified system characteristics.	Section 4.2 Section 4.3	
This analysis shall result in the determination of functional and performance requirements for each primary function of the system. This analysis includes review and update of prior analyses to refine mission and environment definitions to support system definition.	Section 4.4	
Requirements analysis shall be conducted iteratively with functional analysis and synthesis to:	Section 4.2 Section 4.3 Section 4.4 Section 4.5	
(1) Develop requirements that depend on additional system definition (e.g., other elements of the system or performance requirements for identified functions).	Section 4.4	
(2) Verify that product and process solutions can satisfy customer requirements. This analysis shall be used to:	Section 4.5	
• Assist in the refinement of customer objectives and requirements.	Section 4.2	
• Define initial performance objectives and refine them into requirements.	Section 4.2	
• Identify and define constraints that limit solutions (e.g., missions and utilization environments or adverse impacts on natural and human environments).	Section 4.3	
• Select functional and performance requirements based on customer-provided measures of effectiveness. When measures of effectiveness are not provided at the level of detail needed, the contractor shall develop and utilize a set of measures for effectiveness that relate to customer missions; utilization environments; needs, requirements, and objectives; and design constraints.	Section 4.2 Section 4.3	

MIL-STD-499B	*RM Guidebook Paragraph Number*	*Comments*
Functional requirements identified shall be used as the top-level functions for functional analysis. Performance requirements shall be:	Section 4.4	
• Developed iteratively for all identified functions based on system life cycle factors.	Section 4.4 Section 4.5	
• Characterized in terms of the degree of certainty in their estimate, the degree of criticality to system success, and their relationship to other requirements.	Section 4.4 Section 4.5	
4.4 Functional Analysis/Allocation-499B The contractor shall conduct this activity to define and integrate a functional architecture for which system products and processes can be designed.	Section 4.3	
The contractor shall perform functional analysis/allocation to the level of depth needed to support the required synthesis efforts.	Section 4.3	
Identified functional requirements shall be analyzed to determine the lower level functions required to accomplish the parent requirement.	Section 4.3	
All contractually specified usage modes shall be included in the analysis.	Section 4.3	
Functional requirements shall be arranged so that lower level functional requirements are recognized as part of higher level requirements.	Section 4.3 Section 4.4	
Functional requirements shall be arranged in their logical sequence: have their input, output, and functional interface (internal and external) requirements defined and be traceable from beginning to end conditions.	Section 4.3 Section 4.4	
Time critical requirements shall be analyzed.	Section 4.3	
4.4.1 Allocation The contractor shall successively (highest to lowest level) establish performance requirements for each functional requirement and interface.	Section 4.3	
Time requirements that are prerequisite for a function or set of functions shall be determined and allocated. The resulting set of requirements shall be defined in measurable terms and in sufficient detail for use as design criteria.	Section 4.3	
Performance requirements shall be traceable from the lowest level of the current functional architecture, through the analysis by which they were allocated, to the higher level requirements they are intended to fulfill.	Section 4.3 Section 4.4	

MIL-STD-499B	*RM Guidebook Paragraph Number*	*Comments*
4.4.2 Iteration Functional analysis/allocation shall be conducted iteratively:	Section 4.1 Section 4.3 Section 4.4 Section 4.5	
• To define successively lower-level functions required to satisfy higher level functional requirements and to define alternative sets of functional requirements.	Section 4.5	
• With requirements analysis to define mission and environment-driven performance and to determine that higher level requirements are satisfied.	Section 4.2 Section 4.3 Section 4.5	
• To flow down performance requirements and design constraints.	Section 4.4	
• With synthesis to refine the definition of product and process solutions.	Section 4.5	
4.5 Synthesis The purpose of synthesis is to define and design system product and process solutions in terms of design requirements that satisfy the functional architecture and define and integrate the system as a physical architecture. (During concept exploration and definition, the physical architecture should describe the system concept. During demonstration and validation, it should describe the system in terms of its specifications and the concept of Configuration Items (CIs) that make up the system. By the end of engineering and manufacturing development, the physical architecture should provide the detailed design requirements for all system elements and the drawings for the system CIs.)	Section 4.4	
The contractor shall conduct synthesis to a level of detail required to satisfy contract requirements.	Section 4.4	
Synthesis shall be conducted iteratively with:	Section 4.2 Section 4.3 Section 4.4	
(1) Functional analysis/allocation to define a complete set of functional and performance requirements necessary for the level of design output required.	Section 4.3 Section 4.4	

MIL-STD-499B	*RM Guidebook Paragraph Number*	*Comments*
(2) Requirements analysis to verify that solution outputs can satisfy customer input requirements. The contractor shall:	Section 4.2 Section 4.3	
• Define the system, CIs, and system elements for each logical set of functional and performance requirements of the functional architecture.	Section 4.3	
• Determine the completeness of functional and performance requirements for the design.	Section 4.5	
• Define internal and external physical interfaces.	Section 4.4	
• Develop system product and process solutions to a level of detail that (1) enables verification that contractually required accomplishments have been met and (2) provides the basis for the specification tree, Work Breakdown Structure (WBS), and progressive definition of specification and configuration baselines.	Section 4.5	
4.5.1 Design The design outputs from the synthesis shall be complete from an integrated system architecture point of view and they shall:	Section 4.1 Section 4.3 Section 4.4	
• Be the source of data for developing, establishing, and updating (as applicable for the contract effort) the functional, allocated, and product baselines; system, CI, process, and material specifications; drawings and lists; interface control documentation; facility requirements; procedural handbooks and instructional material; and documentation of personal loading.		
• Evidence a simplicity for ease of maintenance (including access and common, noncomplex tools), support (including supply), manufacturing and production (including tooling, layout, processes, and assembly), training, verification, and operational mission accomplishment.		
• Incorporate proven risk reduction methods and templates.		

MIL-STD-499B	*RM Guidebook Paragraph Number*	*Comments*
4.6.1 Trade Studies Desirable and practical trade-offs among stated user requirements, design, program schedule, functional and performance requirements, and life cycle costs shall be identified and executed.	Section 4.3(5)	
Trade-off studies shall be defined, conducted, and documented at the various levels of functional or physical detail to support requirements, functional decomposition/allocation, and design alternative decisions or, as specifically designated, to support the decision needs of the systems engineering process.	Section 4.3(5)	
The level of detail of a trade study shall be commensurate with cost, schedule, performance, and risk impacts.	Section 4.3(5)	
4.6.1.1 Requirements Analysis Trade Studies The contractor shall conduct requirements trade-off studies to:	Section 4.3(5)	
• Establish alternative performance and functional requirements to satisfy customer needs.	Section 4.3(5)	
• Resolve conflicts with customer requirements.	Section 4.3(5)	
4.6.1.2 Functional Analysis/Allocation Trade-Off Studies The contractor shall conduct trade-off studies within and across functions to:	Section 4.1 Section 4.2 Section 4.3(5)	
• Support functional analyses and allocation to performance requirements.		
• Determine the preferred set of performance requirements satisfying identified functional interfaces.		
• Determine performance requirements for lower-level functions when higher level.		
• Evaluate alternative functional architectures.		
4.6.1.3 Synthesis Trade Studies The contractor shall conduct synthesis trade studies to:	Section 4.2 Section 4.3(5)	
• Establish system/CI configuration(s).		
• Assist in selecting system concepts and designs.		
• Support make-or-buy, process, rate, and location decisions.		
• Examine proposed changes.		
• Examine alternative technologies to satisfy functional/design requirements including alternatives for moderate to high risk technologies.		

MIL-STD-499B	*RM Guidebook Paragraph Number*	*Comments*
• Evaluate environmental and cost impacts of materials and processes.		
• Support decisions for new products and process development versus NDI or Commercial Off-the-Shelf (COTS) products and processes.		
• Evaluate alternative physical architectures to select preferred products and processes.		
• Support materials selection.		
• Select standard components, techniques, services, and facilities that reduce system life cycle costs and meet system effectiveness requirements (force structure and infrastructure impacts that emphasize supportability, producibility, training, deployment, and interoperability must be considered).		
4.6.2 System/Cost Effectiveness Analysis	Section 4.1	

11.3 International Standard Operation (ISO) 9000 Requirements

ISO 9000	*RM Guidebook Paragraph Number*	*Comments*
4.4.1 General - 9001 The supplier shall establish and maintain procedures to control and verify the design of the product in order to ensure that the specified requirements are met.	Section 4.5	
4.4.3 Design Input - 9001 Incomplete, ambiguous or conflicting requirements shall be resolved with those responsible for drawing up these requirements.	Section 4.3 Section 4.5	
4.4.4 Design Output - 9001 Design output shall be documented and expressed in terms of requirements, calculations, and analyses. Design output shall:	Section 4.2 Section 4.3 Section 4.4	
• Meet the design input requirements.		
• Contain or reference acceptance criteria.		
• Conform to appropriate regulatory requirements whether or not these have been stated in the input information.		
• Identify those characteristics of the design that are crucial to the safe and proper functioning of the product.		
8.1 Contribution Of Specification And Design To Quality - 9004 The specification and design function should provide for the translation of customer needs from the product brief into technical specifications for materials, products, and processes. This should result in a product that provides customer satisfaction at an acceptable price that enables a satisfactory return on investment for the enterprise.	Section 4.4	
The specification and design should be such that the product or service is producible, verifiable, and controllable under the proposed production, installation, commissioning, or operational conditions.	Section 4.3 Section 4.5	

ISO 9000	*RM Guidebook Paragraph Number*	*Comments*
8.2.2 In its delegation of responsibilities for quality, management should ensure that design functions provide clear and definitive technical data for procurement, the execution of work, and verification of products and processes to specification requirements.	Section 4.1 Section 4.5 Section 4.6	
8.2.4 In addition to customer needs, the designer should give due consideration to the requirements relating to safety, the environment, and other regulations, including items in the company's quality policy which may go beyond existing statutory requirements.	Section 4.3	

11.4 MIL-STD-2167A Requirements

DOD-STD-2167A	RM Guidebook Paragraph Number	*Comments*
4.2.1 Software Development Methods The contractor shall use systematic and well documented software development methods to perform requirements analysis, design, coding, integration, and testing of the deliverable software.	Section 4	Requirements only
The contractor shall implement software development methods that support the formal reviews and audits required by the contract.	Section 4.1 Section 4.6	
4.2.5 Computer Software Organization The contractor shall decompose and partition each CSCI into Computer Software Components (CSCs), and CSUs in accordance with the development method(s) documented in the SDP.	Section 3 Section 4.1	
The contractor shall ensure that the requirements for the CSCI are completely allocated and further refined to facilitate the design and test of each CSC and CSU.	Section 4.2 Section 4.3 Section 4.4	
4.2.3 Safety Analysis The contractor shall perform the analysis necessary to ensure that the software requirements, design, and operating procedures minimize the potential for hazardous conditions during the operational mission.	Section 4.3	
Any potentially hazardous conditions or operating procedures shall be clearly identified and documented.	Section 4.4	
4.2.6 Traceability of Requirements to Design The contractor shall document the traceability of the requirements allocated from the system specification to each CSCI, its CSCs and CSUs, and from the CSU level to the SRSs and IRSs.	Section 4.3 Section 4.5	
4.2.10 Processing Resource and Reserve Capacity The contractor shall analyze the processing resource and reserve requirements, such as timing, memory utilization, Input/Output (I/O) channel utilization, identified in the contract and shall allocate these resources among the CSCIs.	Section 4.2	
The allocation of these resources to a CSCI shall be documented in the SRS for that CSCI.	Section 4.4	
The contractor shall monitor the utilization of processing resources for the duration of the contract and shall reallocate the resources as necessary to satisfy reserve requirements.	Section 4.1 Section 4.6	

DOD-STD-2167A	*RM Guidebook Paragraph Number*	*Comments*
5.1 System Requirements Analysis/Design The contractor shall perform the following system requirements analysis/design activities.	Section 4	
5.1.1 Software Development Management		
5.1.1.1 The contractor shall support the System Requirements Reviews (SRR) as specified in the contract.	Section 4.1 Section 4.6	
5.1.1.2 The contractor shall support the System Design Review (SDR) as specified in the contract.	Section 4.1 Section 4.6	
5.1.2.1 The contractor shall analyze the preliminary system specification and shall determine whether the software requirements are consistent and complete.	Section 4.2	
5.1.2.2 The contractor shall conduct analysis to determine the best allocation of system requirements between hardware, software, and personnel in order to partition the system into HWCIs, CSCIs, and manual operations.	Section 4.3	
The contractor shall document the allocation of system requirements in the SSDD.	Section 4.4	
5.1.2.3 The contractor shall define a preliminary set of engineering requirements for each CSCI.	Section 4.2 Section 4.3	
The contractor shall document the preliminary set of engineering requirements in the preliminary SRS for each CSCI.	Section 4.4	
5.1.2.4 The contractor shall define a preliminary set of interface requirements for each interface external to each CSCI.	Section 4.2 Section 4.3	
The contractor shall document these requirements in a preliminary IRS.	Section 4.4	

DOD-STD-2167A	*RM Guidebook Paragraph Number*	*Comments*
5.1.3 The contractor shall define a preliminary set of qualification requirements for each CSCI.	Section 4.2 Section 4.3	
The contractor shall document these requirements in the preliminary SRS for each CSCI. These requirements shall be consistent with the qualification requirements defined in the system specification.	Section 4.4	
5.2 Software Requirements Analysis The contractor shall perform the following software requirements analysis activities.	Section 4	
5.2.1 Software Development Management The contractor shall conduct one or more Software Specification Reviews (SSR) in accordance with MIL-STD-1521.	Section 4.1 Section 4.6	
Following successful completion of an SSR and when authenticated by the contracting agency, the SRSs and associated IRS will establish the allocated baseline for the CSCI.	Section 4.5 Section 4.6	
5.2.2.1 The contractor shall define a complete set of engineering requirements for each CSCI.	Section 4.2 Section 4.3	
The contractor shall establish these requirements in the SRS for each CSCI.	Section 4.4	
5.2.2.2 The contractor shall define a complete set of interface requirements for each interface external to the CSCI.	Section 4.2 Section 4.3	
The contractor shall document these requirements in an IRS.	Section 4.4	
5.2.3 Formal Qualification Testing (FQT) The contractor shall define a complete set of qualification requirements for each CSCI.	Section 4.1 Section 4.2 Section 4.3	
The contractor shall document these requirements in the SRS for each CSCI.	Section 4.4	
5.4.2.1 The contractor shall develop a detailed design for each CSCI, shall allocate requirements from the CSCs to the CSUs of each CSCI, and shall establish design requirements for each CSU.	Section 4.2 Section 4.3	
The contractor shall document this information in the SDD for each CSCI.	Section 4.4	

This page left intentionally blank

11.5 DOD Instruction (DODI) 5000.2 Requirements

DODI 5000.2 Part 6	*RM Guidebook Paragraph Number*	*Comments*
3.b.(1).(a) In the broadest sense, the systems engineering process begins when either the need for a capability is recognized or the opportunity to exploit a technology presents itself and is converted into defined operational requirements. These requirements are further translated into detailed design specifications.	Section 4.1 Section 4.2	
3.b.(1).(b) The program office will work with the user or user's representative to establish feasible operational requirements and identify the critical operational characteristics and constraints.	Section 4.1 Section 4.2 Section 4.5	
3.b.(1).(b).1 A disciplined requirements collection and translation methodology will be used to convert these requirements into detailed design specifications.	Section 4.1	
3.c.(2) The systems engineering process will allocate system requirements to establish clear technical requirements for each technical specialty in a concurrent manner to support the integrated system design.	Section 4.3	
The systems engineering process will collectively analyze the design specifications, conduct trade-offs, balance total system requirements, and establish the final configuration.	Section 4.3	

This page left intentionally blank

11.6 MIL-STD-498 Requirements

MIL-STD-498	*RM Guidebook Paragraph Number*	*Comments*
4.1 Software Development Process The software development process shall include the following major activities, which may overlap, may be applied iteratively, may be applied differently to different elements of software, and need not be performed in the order listed.	Section 4	
• System requirements analysis.	Section 4.1 Section 4.2	
• Software requirements analysis.	Section 4.3	
The developer's software development process shall be described in the SDP.	Section 4.1	
4.2.4 Handling of Critical Requirements The developer shall meet the following requirements	Section 4.3	
4.2.4.1 Safety Assurance The developer shall identify as safety-critical those CSCIs or portions thereof whose failure could lead to a hazardous system state (one that could result in unintended death, injury, loss of property, or environmental harm).	Section 4.3	
If there is such software, the developer shall develop a safety assurance strategy including both tests and analyses, to assure that the requirements, design, implementation, and operating procedures for the identified software minimize or eliminate the potential for hazardous conditions.		
The strategy shall include a software safety program that shall be integrated with the system safety program, if one exists.		
The developer shall: • Record the strategy in the SDP.	Section 4.1 Section 4.3	
• Implement the strategy.	Section 4.4	
• Produce evidence, as part of required software products, that the safety assurance strategy has been carried out.	Section 4.5	

MIL-STD-498	*RM Guidebook Paragraph Number*	*Comments*
4.2.4.2 Security Assurance The developer shall identify as security-critical those CSCIs or portions thereof whose failure could lead to a breach of system security.	Section 4.3	
If there is such software, the developer shall develop a security assurance strategy to assure that the requirements, design, implementation, and operating procedures minimize or eliminate the potential for breaches of system security.		
The developer shall: • Record the strategy in the SDP.	Section 4.1 Section 4.3	
• Implement the strategy.	Section 4.4	
• Produce evidence, as part of required software products, that the security assurance strategy has been carried out.	Section 4.5	
4.2.4.3 Privacy Assurance The developer shall identify as privacy-critical those CSCIs or portions thereof whose failure could lead to a breach of system privacy.	Section 4.3	
If there is such software, the developer shall develop a privacy strategy to assure that the requirements, design, implementation, and operating procedures for the identified software minimize or eliminate the potential for breaches of system privacy.		
The developer shall perform the following: • Record the strategy in the SDP.	Section 4.1 Section 4.3	
• Implement the strategy.	Section 4.4	
• Produce evidence, as part of required software products, that the privacy assurance strategy has been carried out.	Section 4.5	
4.2.4.4 Assurance of Other Critical Requirements If a system relies on software to satisfy other requirements deemed critical by the contract or by system specification, the developer shall identify those CSCIs or portions thereof whose failure could lead to violation of those critical requirements.	Section 4.3	

MIL-STD-498	*RM Guidebook Paragraph Number*	*Comments*
If there is such software, the developer shall develop a strategy to assure that the requirements, design, implementation, and operating procedures for the identified software minimize or eliminate the potential for such violations.		
The developer shall: • Record the strategy in the SDP.	Section 4.1 Section 4.3	
• Implement the strategy.	Section 4.4	
• Produce evidence, as part of required software products, that the assurance strategy has been carried out.	Section 4.5	
4.2.5 Computer Hardware Resource Utilization The developer shall analyze contract requirements concerning computer hardware resource utilization (such as maximum allowable use of processor capacity, memory capacity, I/O device capacity, auxiliary storage devices capacity, and communications/network equipment capacity).	Section 4.3	
The developer shall allocate computer hardware resources among the CSCIs.	Section 4.3	
The developer shall monitor the hardware resource utilization for the duration of the contract.	Section 4.3	
The developer shall reallocate or identify the need for additional hardware resources as necessary to meet contract requirements.	Section 4.3	
4.2.6 Recording Rationale The developer shall record the rationale that will be useful to the support agency for key decisions made in specifying, designing, implementing, and testing the software. The meaning of "key decisions" and the approach for providing the rationale shall be described in the SDP.	Section 4.4	
This rationale shall include: • Trade-offs considered.		
• Analysis methods.		
• Criteria used to make the decisions.		
The rationale shall be recorded in: • Documents.		
• Code comments.	Section 4.3	
• Other media that will transition to the support agency.	Section 4.3 Section 4.4	

MIL-STD-498	*RM Guidebook Paragraph Number*	*Comments*
5.3 System Requirements Analysis The developer shall participate in system requirements analysis in accordance with the following requirements.	Section 4.0	
If a system is developed in multiple builds, its requirements may not be fully defined until the final build. The developer's planning should identify the subset of requirements to be defined in each build.	Section 4.3	
The developer's planning should identify the subset of requirements to be implemented in each build. System requirements analysis for a given build should be interpreted to mean defining the system requirements so identified for that build.	Section 4.3 Section 4.4	
5.3.1 Analysis of User Input The developer shall participate in analyzing user input provided by the acquirer to gain an understanding of user needs. This input may take the form of need statements, surveys, problem/change reports, feedback on prototypes, interviews, or other user input or feedback.	Section 4.2	
5.3.2 Operational Concept The developer shall participate in defining and recording the operational concept for the system. The result shall include all applicable items in the OCD and DID.	Section 4.1	
5.3.3 System Requirements The developer shall participate in defining and recording the requirements to be met by the system and the methods to be used to ensure that each requirement has been met.	Section 4.2	
The resulting documentation shall include all applicable items in the SSS DID. Depending on Contract Data Requirements List (CDRL) provisions, requirements concerning system interfaces may be included in the SSS or in IRSs. Note: If a system consists of subsystems, the activity in 5.3.3 is intended to be performed iteratively with the activities in 5.4 (system design) to define system requirements, design the system and identify its subsystems, define the requirements for those subsystems, design the subsystems and identify their components, and so on.	Section 4.4	
5.4.1 System-wide Design Decisions The developer shall participate in defining and recording system-wide design decisions (that is, decisions about the system's behavioral design and other decisions affecting the selection and design of system components).	Section 4.4	

MIL-STD-498	*RM Guidebook Paragraph Number*	*Comments*
5.4.2 System Architectural Design The developer shall participate in defining and recording the architectural design of the system (identifying the components of the system, their interfaces, and a concept of execution among them) and the traceability between the components and system requirements.	Section 4.3	
5.5 Software Requirements Analysis The developer shall define and record the software requirements to be met by each CSCI.	Section 4.3 Section 4.4	
The developer shall define and record the methods to be used to ensure that each requirement has been met.	Section 4.3 Section 4.4	
The developer shall define and record the traceability between the CSCI requirements and system requirements.	Section 4.3 Section 4.4	
5.6.1 CSCI-wide Design Decisions The developer shall define and record CSCI-wide design decisions (that is, decisions about the CSCI's behavioral design and other decisions affecting the selection and design of the software units comprising the CSCI).	Section 4.4	
5.6.2 CSCI Architectural Design The developer shall define and record the architectural design of each CSCI (identifying the software units comprising the CSCI, their interfaces, and a concept of execution among them).	Section 4.4	
The developer shall define and record the traceability between the software units and the CSCI requirements.	Section 4.3 Section 4.4	
5.9 CSCI Qualification Testing The developer shall perform CSCI qualification testing in accordance with the following requirements: Note 1: CSCI qualification testing is performed to demonstrate to the acquirer that CSCI requirements have been met. It covers the CSCI requirements in SRSs and in associated IRSs. This testing contrasts with developer-internal CSCI testing which is performed as the final stage of unit integration and testing. Note 2: If a CSCI is developed in multiple builds, its CSCI qualification testing will not be completed until the final build for that CSCI or possibly until later builds involving items with which the CSCI is required to interface. CSCI qualification testing in each build should be interpreted to mean planning and performing tests of the current build of each CSCI to ensure that the CSCI requirements to be implemented in that build have been met.	Section 4.5 Section 4.6	

MIL-STD-498	*RM Guidebook Paragraph Number*	*Comments*
5.9.3 Preparing for CSCI Qualification Testing The developer shall define and record the test preparations, test cases, and test procedures to be used for CSCI qualification testing.	Section 4.4 Section 4.5	
The developer shall define and record the traceability between the test cases and the CSCI requirements.	Section 4.4 Section 4.5	
The resulting documentation shall include all applicable items in the Software Test Description (STD) DID.	Section 4.4 Section 4.5	
The developer shall prepare the test data needed to carry out the test cases and provide the acquirer advance notice of the time and location of CSCI qualification testing.	Section 4.4 Section 4.5	
5.11 System Qualification Testing The developer shall participate in system qualification testing in accordance with the following requirements: Note 1: System qualification testing is performed to demonstrate to the acquirer that system requirements have been met. It covers the system requirements in the SSSs and in associated IRSs. This testing contrasts with developer-internal system testing which is performed as the final stage of CSCI/HWCI integration and testing. Note 2: If a system is developed in multiple builds, qualification testing of the completed system will not occur until the final build. System qualification testing in each build should be interpreted to mean planning and performing tests of the current build of the system to ensure that the system requirements to be implemented in that build have been met.	Section 4.5 Section 4.6	
5.11.3 Preparing for System Qualification Testing The developer shall participate in developing and recording the test preparations, test cases, and test procedures to be used in system qualification testing.	Section 4.4 Section 4.5	
The developer shall define and record the traceability between the test cases and the system requirements.	Section 4.4 Section 4.5	
For software systems, the resulting documentation shall include all applicable items in the STD DID.	Section 4.4 Section 4.5	
The developer shall participate in preparing test data needed to carry out the test cases and in providing the acquirer advance notice of the time and location of system qualification testing.	Section 4.4 Section 4.5	

MIL-STD-498	*RM Guidebook Paragraph Number*	*Comments*
5.13.4 Preparing the "As Built" CSCI Design and Related Information The developer shall update the design description of each CSCI to match the "as built" software and define and record…	Section 4.3 Section 4.4	
• The traceability between the CSCI's source files and software units.	Section 4.3 Section 4.4	
• Traceability between the computer hardware resource utilization measurements and the CSCI requirements concerning them.	Section 4.3 Section 4.4	
The resulting documentation shall include all applicable items in the qualification, software support, and traceability sections of the SPS DID.	Section 4.5	

This page left intentionally blank

11.7 EIA 632 Requirements

EIA 632	*RM Guidebook Paragraph Number*	*Comments*
4.1.1 Supply Process - Agreement Satisfaction For a system, or portion thereof, supplied to an acquirer, the developer (when acting as the supplier) shall establish and satisfy an agreement with the acquirer.	Section 4.1 Section 4.5 Section 4.6	
4.1.2 Acquisition Process - Product Acquisition For a system, or portion thereof, acquired from a supplier, the developer (when acting as the acquirer) shall establish an agreement with that supplier.	Section 4.1	
4.1.2 Acquisition Process - Supplier Performance The developer (when acting as the acquirer) shall manage supplier performance to ensure that the technical effort to be accomplished by the supplier provides end products that satisfy the assigned requirements.	Section 4.5 Section 4.6	
4.2.1 Planning Process - Implementation Strategy The developer shall define a strategy for implementing the adopted processes as a basis for project technical planning and that is in accordance with the agreement.	Section 4.1	
4.2.1 Planning Process - Technical Effort Definition The developer shall define a technical effort that is in accordance with the implementation strategy.	Section 4.1 Section 4.2	
4.2.1 Planning Process - Schedule and Organization The developer shall schedule and organize the defined technical effort.	Section 4.1 Section 4.4	
4.2.1 Planning Process - Technical Plans The developer shall create technical plans to ensure an integrated and cost effective technical effort in accordance with the defined schedule and organization.	Section 4.1	
4.2.1 Planning Process - Work Directives The developer shall create work directives that implement the planned technical effort.	Section 4.1 Section 4.4	
4.2.2 Assessment Process - Progress Against Plans and Schedules The developer shall assess the progress of the technical effort against applicable technical plans and schedules.	Section 4.1 Section 4.6	

EIA 632	*RM Guidebook Paragraph Number*	*Comments*
4.2.2 Assessment Process - Progress Against Requirements The developer shall assess the progress of system development by comparing currently defined system characteristics against requirements.	Section 4.5 Section 4.6	
4.2.2 Assessment Process - Technical Reviews The developer shall conduct technical reviews of progress and accomplishments in accordance with appropriate technical plans.	Section 4.1 Section 4.6	
4.2.3 Control Process - Outcomes Management The developer shall manage the outcomes of the technical effort.	Section 4.1 Section 4.3 Section 4.5	
4.2.3 Control Process - Information Dissemination The developer shall ensure that required and requested information is disseminated in accordance with the agreement, project plans, enterprise policies, and enterprise procedures.	Section 4.1 Section 4.2 Section 4.5 Section 4.6	
4.3.1 Requirements Definition Process - Acquirer Requirements The developer shall define a validated set of acquirer requirements for the system, or portion thereof.	Section 4.4	
4.3.1 Requirements Definition Process - Other Stakeholder Requirements The developer shall define a validated set of other stakeholder requirements for the system, or portion thereof.	Section 4.4	
4.3.1 Requirements Definition Process - System Technical Requirements The developer shall define a validated set of system technical requirements.	Section 4.4	
4.3.2 Solution Definition Process - Logical Solution Representations The developer shall define one or more validated sets of logical solution representations that conform with the technical requirements of the system.	Section 4.4	
4.3.2 Solution Definition Process - Physical Solution Representations The developer shall define a preferred set of physical solution representations that is in accordance with the assigned logical solution representations, derived technical requirements, and system technical requirements.	Section 4.4	
EIA 632	*RM Guidebook Paragraph Number*	*Comments*

4.3.2 Solution Definition Process - Specified Requirements The developer shall specify requirements for the design solution of the preferred physical solution representation.	Section 4.4	
4.4.1 Implementation Process The developer shall implement the design solution in accordance with the specified requirements to obtain a verified end product.	Section 4.5	
4.4.2 Transition to Use Process The developer shall transition verified products to the acquirer of the products in accordance with the agreement.	Section 4.6	
4.5.1 Systems Analysis Process - Effectiveness Analysis The developer shall perform effectiveness analyses to provide a quantitative basis for decision making.	Section 4.3	
4.5.1 Systems Analysis Process - Tradeoff Analysis The developer shall perform tradeoff analyses to provide decision makers with recommendations, predictions of the results of alternative decisions, and other appropriate information to allow selection of the best course of action.	Section 4.2	
4.5.1 Systems Analysis Process - Risk Analysis The developer shall perform risk analyses to develop risk management strategies, support management of risks, and support decision making.	Section 4.1 Section 4.3	
4.5.2 Requirements Validation Process - Requirement Statements The developer shall ensure that technical requirement statements and specified requirement statements, individually and as sets, are stated in an acceptable manner.	Section 4.5	
4.5.2 Requirements Validation Process - Acquirer Requirements The developer shall ensure that the set of defined acquirer requirements conforms to acquirer needs and expectations.	Section 4.3	
4.5.2 Requirements Validation Process - Other Stakeholder Requirements The developer shall ensure that the set of defined other stakeholder requirements conforms to other stakeholder needs and expectations with respect to the system.	Section 4.3	

EIA 632	*RM Guidebook Paragraph Number*	*Comments*
4.5.2 Requirements Validation Process - System Technical Requirements The developer shall ensure that the set of defined system technical requirements conforms to the validated acquirer and other stakeholder requirements.	Section 4.5	
EIA 632	*RM Guidebook Paragraph Number*	*Comments*
4.5.2 Requirements Validation Process - Logical Solution Representations The developer shall ensure that each set of logical solution representations conforms to the appropriately assigned subset of system technical requirements.	Section 4.5	
4.5.3 Product Verification Process - Design Solution The developer shall verify that end products defined by the system design solution conform to source requirements.	Section 4.5	
4.5.3 Product Verification Process - End Product The developer shall verify that an end product to be delivered to an acquirer conforms to its specified requirements.	Section 4.5	
4.5.3 Product Verification Process - Enabling Product Readiness The developer shall determine readiness of enabling products for development, production, test, deployment/installation, training, support/maintenance, and retirement or disposal.	Section 4.5 Section 4.6	
4.5.4 Product Validation Process The developer shall ensure that an end product received from a supplier, or to be delivered to an acquirer, conforms to the acquirer requirements associated with that end product.	Section 4.5 Section 4.6	

Software Requirements Management Key Process Area Document Change Request Form

This form is designed to allow you to send change requests to the editor. Send a filled out copy of this form to:

PRC Inc.
Attn: K. Ptack
21789 North Coral Drive, Suite 1A
Lexington Park, MD 20653-1039

FAX Number: (301) 862-3850

E-mail Address: _____
Name of Submitted Organization: _____
Organization Contact/Sender's Name: _____
Telephone: _____
Address: _____
_____
_____
Short Descriptive Title of the Change: _____
Title of Document: _____
Location: _____
(page #, section #, paragraph #, sentence #)

PROPOSED CHANGE:

_____
_____
_____
_____
_____
_____
_____
_____

RATIONALE FOR CHANGE:

_____
_____
_____
_____
_____
_____
_____
_____

Note 1: For the SRM KPA WG to take appropriate action on a change request, we must have a clear description of the recommended change along with a supporting rationale.