Department of Energy (DOE)

Systems Engineering Methodology Version 3

The DOE Systems Development Lifecycle (SDLC) for Information Technology Investments

September 2002

U. S. DEPARTMENT OF ENERGY
Office of the Chief Information Officer

TITLE PAGE

Document Name:	Department of Energy Systems Engineering Methodology (SEM) *The DOE Systems Development Lifecycle (SDLC) for Information Technology Investments*
Publication Date:	
Original	March 1996; Revised November 1997
Version 2	March 1999
Version 3	September 2002

Approval:

_____

Karen Evans, Chief Information Officer

_____

Brenda Coblentz, Program Manager,
Software Quality & Systems Engineering

SEM Version 1 became a DOE directive on May 21, 1997 as DOE G 200.1-1.
Version 3 will become directive number DOE G 200.1-1A.

Notice: This document and associated process guides can be found at Web site
http://cio.doe.gov/ITReform/sqse

U.S. DEPARTMENT OF ENERGY
Office of the Chief Information Officer

Preface

The initial development and ongoing revisions of the *Department of Energy Systems Engineering Methodology (SEM)*, first published in March 1996, are performed as part of a continuing effort to improve the quality, performance, and productivity of Departmental information systems. DOE Federal and contractor personnel are involved in the evolution of the document as contributors or reviewers.

The key changes for Version 3 include alignment to all of the Level 3 key process areas of the Software Capability Maturity Model (CMM-SW) developed by the Software Engineering Institute at Carnegie Mellon University, and revisions to bring it up to date with guidance changes in the DOE Information Technology (IT) environment since the 1999 update (version 2). The revisions were reviewed by a SEM Change Control Board composed of members of the DOE Software Quality & Systems Engineering staff. To summarize, the major enhancements include:

- A renaming of the *Department of Energy Software Engineering Methodology (SEM)* to the *Department of Energy Systems Engineering Methodology (SEM), the DOE Systems Development Lifecycle (SDLC) for Information Technology Investments.* This includes revisions throughout to focus not only on software but all IT projects.
- A mapping of the SEI CMM-SW Level 2 and 3 criteria to the SEM, available on the Software Quality & Systems Engineering web site.
- Renaming and updating of the Programming Stage to the Construction Stage.
- Revision of the planning questionnaire for a “total systems approach” to a project.
- Repagination and combination of the SEM files to enhance electronic reading of the document.
- The conversion of the appendixes for Structured Walkthroughs, In-Stage Assessments, and Stage Exists from SEM appendixes to stand-alone documents available on the Software Quality & Systems Engineering web site.
- Greater emphasis on risk management and performance measures
- Updated information on the DOE Enterprise Architecture.

Chapter Page

Chapter Page

Chapter Page

Chapter Page

Exhibits **Page**

Web Site	Address[1]

DOE Sites

Software Quality & Systems Engineering ..... http://cio.doe.gov/ITReform/sqse

CIO Web Site ..... http://cio.doe.gov

Corporate Information Management Guidance Item H1002 *DOE Systems Engineering for Information Systems* ..... http://www-it.hr.doe.gov/implan/Corporate_Guidance/cimg898.htm#H1002

DOE Directives ..... http://www.directives.doe.gov

DOE IT Standards ..... http://cio.doe.gov/ITReform/ArchitectureStandards/ASP.html

DOE Standards ..... http://www.osti.gov/

DOE Cyber Security Program ..... http://cio.doe.gov/Cybersec/index.html

Headquarters Computer Security Program ..... http://cio.doe.gov/Cybersec/index.html

Information Architecture Program ..... http://cio.doe.gov/ITReform/ArchitectureStandards/ASP.html

Records Management ..... http://cio.doe.gov/RBManagement/Records/records.html

Strategic Information Management (SIM) ..... http://cio.doe.gov/ITReform/SIM

Software Risk Management: A Practical Guide ..... http://cio.doe.gov/sqas/publications.htm

SQ&SE Web Site, Risk Page ..... http://cio.doe.gov/ITReform/sqse/pm_risk.htm

Federal Government Sites

OMB on Privacy ..... http://www.whitehouse.gov/omb/privacy/general.html

Industry Sites

Carnegie Mellon University Software Engineering Institute ..... http://www.sei.cmu.edu

International Council on Systems Engineering ..... http://www.incose.org

Project Management Institute ..... http://www.pmi.org

Quality Assurance Institute ..... http://www.qaiusa.com

[1] Addresses are as of December 2002

Chapter: **1.0 Introduction**

Description: The *Department of Energy Systems Engineering Methodology (SEM)* provides guidance for information systems[1] engineering, project management, and quality assurance practices and procedures. The primary purpose of the methodology is to promote the development of reliable, cost-effective, computer-based solutions while making efficient use of resources. Use of the methodology will also aid in the status tracking, management control, and documentation efforts of a project.

This information systems engineering methodology is consistent with other methodologies used in the Government and private industry. It complies with Departmental policy on project management, configuration management, security, and records management. It should be used in conjunction with Departmental information management programs and initiatives.

Significant input for the methodology was obtained from information management programs at sites and organizations throughout the Department. The methodology integrates Departmental best practices and focuses on the quality of both the systems engineering process and the work products generated from the process.

The SEM is derived from the principles and standards advocated by information management industry leaders, such as The Institute of Electrical and Electronics Engineers (IEEE) and the Carnegie Mellon Software Engineering Institute (SEI). This methodology is designed to enable project teams to fully achieve Level 3 maturity on the SEI Capability Maturity Model.

Quality assurance is integrated into the methodology, making quality the responsibility of all project team manager(s) and members. To assure the development of quality products, the methodology prescribes reviews, inspections, and audits for the lifecycle processes and technical work products. To protect the integrity of information systems, the methodology also prescribes configuration controls over system components, data, and technical documentation.

The methodology encompasses all aspects of the information systems engineering project lifecycle, from project planning through production and maintenance, and integrates the following basic lifecycle management concepts.

1 As defined by Office of Management and Budget Circular No. A-130, Management of Federal Resources, 11/28/2000, the term "information system" means a discrete set of information resources organized for the collection, processing, maintenance, transmission, and dissemination of information, in accordance with defined procedures, whether automated or manual. Within the context of the SEM, the term includes the infrastructure that supports information systems development and operations.

Description, continued:

- Implementation of information systems engineering preferred practices using a graded approach based on the level of effort, complexity, and degree of external impact of the solution.

- Implementation of a project management methodology including quality assurance, configuration management, and a comprehensive testing approach that is adaptable to the individual DOE site environments.

- Application of a complete documentation approach supporting both lifecycle and project management activities, to assure an effective method for managing, tracking, and evaluating information systems engineering activities.

The SEM is intended to be used by individuals, project teams, and managers who are responsible for developing a new computer-based solution or effecting changes to an existing system. The methodology is reviewed on a regular basis and will be modified as needed to keep pace with the changing needs of the Departmental information systems engineering environment and the continuing technical advances in the information systems industry.

The following sections provide additional information about using the SEM.

Section: **1.1 Relationship of the SEM to Other CIO and DOE Corporate Programs**

Description: The SEM defines the DOE Systems Development Lifecycle (SDLC) for information technology (IT) investments (projects) and is managed and maintained by the Office of the CIO. There are a number of ancillary activities that are performed throughout the SDLC by various DOE organizations in support of the successful execution of IT projects. The additional activities are defined by other DOE CIO and corporate programs and are designed to meet policies and procedures, requirements imposed by legislation, including the Paperwork Reduction Act and the Clinger-Cohen Act, and the priorities established by the Secretary. These activities are a part of, but are not limited to, the following processes:

- Departmental Management Processes:
  DOE Strategic Planning Process
  DOE Budget process
  DOE Procurement Process
  DOE Capital Asset Management Process

- Core Information Technology Management processes:
  IT Enterprise Architecture Process
  IT Capital Planning and Investment Control Process
  IT Project Management Process

- Information Technology Control Processes
  Data Management Process
  IT Standards Management Process

Process Evolution: This section briefly describes the components of an integrated IT management framework. At this issuance, each specific component process of the DOE framework is at varying levels of maturity and is continuously evolving towards a fully integrated state. These processes create data about DOE IT investments and projects, which, when created and provided on a regular and timely basis, can be a significant asset to assist management in decision-making.

Process Descriptions:

DOE Strategic Planning Process. The purpose of strategic planning is to ensure that through effective preparation, DOE programs and support activities are positioned to achieve long-term Departmental goals and objectives. Strategic planning assists the Secretary, Deputy Secretary, and Under Secretaries in setting the long-term directions and policies for the Department and in making decisions on near-term priorities and resource allocations.

The Departmental Strategic Plan is the foundation for all DOE planning, budgeting, execution, control, and evaluation activities by Program Offices and support organizations. The Department's long-term goals, objectives, priorities, and performance measures are defined, agreed-to, and published in the form of the DOE Strategic Plan, Program Strategic Plan, Annual Performance Plan, and the Information Resource Management (IRM) Strategic Plan.

The SEM provides guidance for ensuring that projects have been planned and approved through the established planning process.

DOE Budget Process. The DOE budget formulation process is the vehicle by which programs and projects are proposed and justified for funding. The DOE budget execution process ensures that funds are used and expended for approved purposes. The Department's budget process stretches over a period of approximately 1.5 years and can be longer if appropriations are not enacted by September 30. The budget process has four distinct formulation phases: field budget, corporate review budget, Office of Management and Budget, and the Congressional Budget.

Certain major IT projects are specifically included in the DOE budget, while for other projects information is gathered and submitted as an attachment to the budget. The SEM provides guidance to ensure that projects are approved by stakeholders responsible for the DOE budget process.

DOE Procurement Process. Consistent with Federal procurement and acquisition regulations, DOE has a procurement process and organization to ensure that Federal dollars spent to acquire assets or services are properly justified, documented, and expended. There are basic rules and regulations common to most DOE procurements, including those for IT products and services.

The *Department of Energy Acquisition Regulations* (DEAR) and the *Federal Acquisition Regulations* (FAR) govern the DOE procurement process. The FAR and the DEAR recognize that IT procurements require special rules and set forth such unique rules in FAR Part 39 and DEAR Part 939. The SEM provides guidance for the development and approval of a project acquisition plan.

Process Descriptions, continued:

DOE Capital Asset Management Process. DOE Policy P 413.1 and DOE Order O 413.3 provide project management direction for the planning, budgeting, acquisition, and construction of capital assets. Capital assets are defined as land, structures, equipment, and information technology assets that have a useful life of two years or more. A Program and Project Management Manual that includes specific detailed requirements for managing major systems engineering projects was published in February 2002.

The capital asset management process identifies critical decisions that must be made as a project progresses through its lifecycle. A critical decision is a formal determination or decision at a specific point in a project stage that allows the project to continue to the next stage and commits resources. The SEM defines all project activities for full compliance with DOE Order 413 and the Program and Project Management Manual, as well as industry and government best practices.

IT Enterprise Architecture Process. An information technology (IT) enterprise architecture (EA) is the explicit description and documentation of the current and desired relationships among business and management processes and the technology that supports the processes. An EA describes the current and target architectures, as well as the transition strategy. The EA strengthens management of the Department's information and the effective use of it.

The Department's EA addresses at a high level the business functions and data required to perform DOE key business operations (see the Information Architecture Program web site for current provisions). It also summarizes the significant applications that support DOE current and future business functions and data, and characterizes the technology infrastructure. Program and site specific functions are addressed by Program Office and site architectures and solutions. The SEM includes guidance to ensure that projects are aligned with the target DOE architecture.

IT Capital Planning and Investment Control Process. To maximize the return to the Department on its substantial investment in IT resources, a structured IT capital planning and investment control (CPIC) process is used to govern IT investments. Under this process, each ongoing or proposed IT investment is subject to consistent selection criteria including a business case, control mechanisms, and evaluation to ensure that all IT investments are justified and well managed. In addition, the CPIC process allows DOE to view its IT investments as a portfolio of projects ensuring an appropriate balance among infrastructure, administrative, and scientific IT investments.

This systematic approach enables DOE to achieve the maximum organizational benefit and reduce IT redundancy. The CPIC process includes three components: Selection, Control, and Evaluation. The SEM provides complete guidance for

Process Descriptions, continued:

managing all aspects of the Control component of the CPIC process, including analysis of benefits and costs (business case), as well as system maintenance in Evaluation. *Exhibits 1.1-1 – 1.1-3, Capital Planning and Investment Control Processes*, illustrate the SEM relationship to the CPIC framework. They also illustrate key components, activities, and work products of the SEM.

IT Project Management Process. DOE recognizes the importance of high quality, systematic project management as a key success factor in the accomplishment of planned project objectives and the realization of projected benefits. Project management has two tightly linked components: a business and a technical component. The business component focuses on project initiation and justification, project planning and control, and project evaluation and closeout. The technical component deals with requirements definition, technical design, acquisition or development, testing, installation, and operation of the system. The SEM provides guidance for addressing all project management activities throughout each stage of the project.

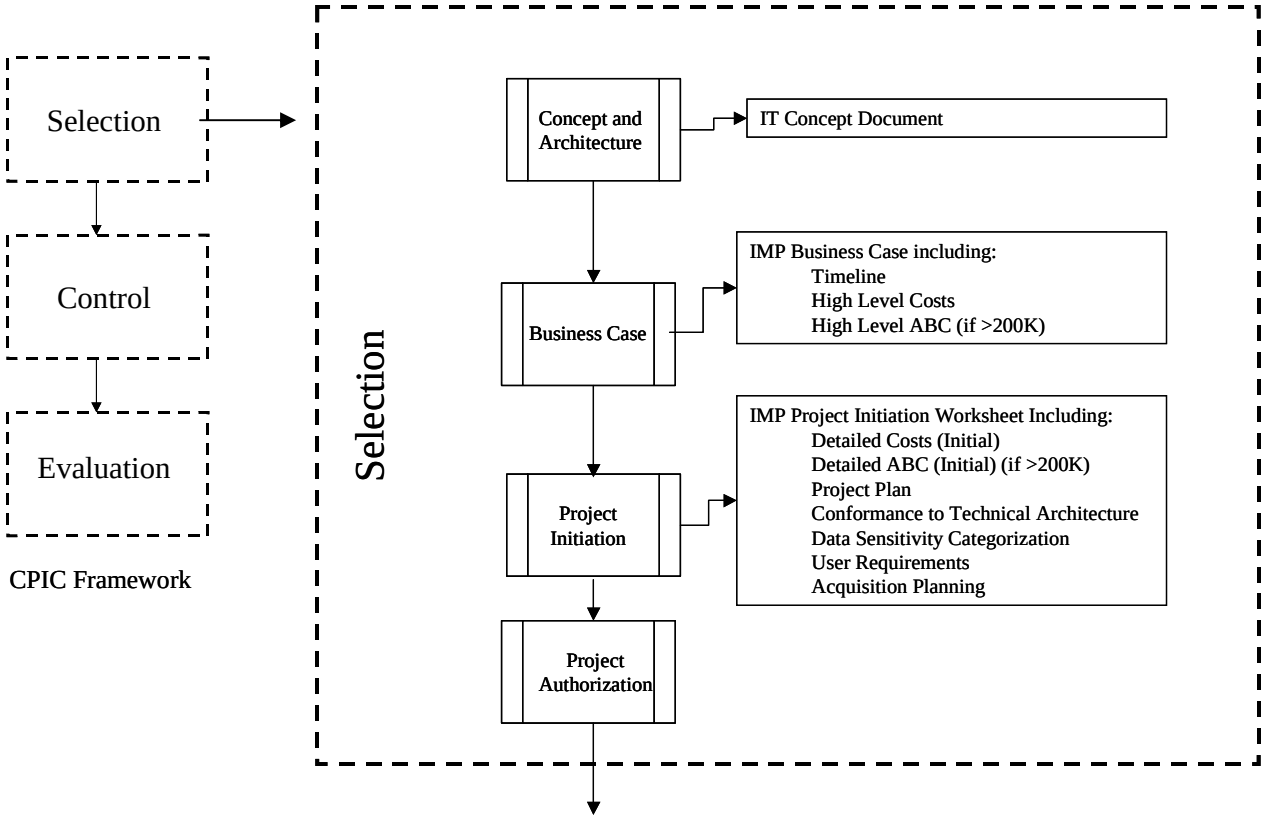
Data Management Process. A data management process implements and manages DOE data in such a way that all data resources are employed as efficiently as possible in support of the DOE mission and operational business objectives. It allows for DOE data to be accessible, credible, and usable, and provides a framework of data resources available and responsive to the functional areas within DOE. A data management process creates and maintains the enterprise data model and drives the DOE data management function. It supports activities needed to comply with the Office of Management and Budget (OMB) Data Quality Guidelines. The SEM addresses data management activities throughout the project lifecycle.

IT Standards Management Process. The purpose of the IT standards management process is to establish and maintain an IT standards profile that supports DOE IT services. The DOE *Profile of Adopted Standards* reflects DOE-wide consensus on standards supporting the DOE technology service areas as defined by the technical reference model (TRM). The *Profile* integrates international, national, federal, and industry standards. The IT Standards Profile includes a security standards profile specific to DOE IT security services.

The OCIO has established a management plan for the IT Standards management process which includes IT standards development, the IT Standards Adoption and Retirement process, the use of IT standards by DOE Program Offices and sites, and the administration and management of the IT Standards management process. The SEM provides guidance that is aligned with the DOE IT standards and advocates use of the standards in all information systems projects.

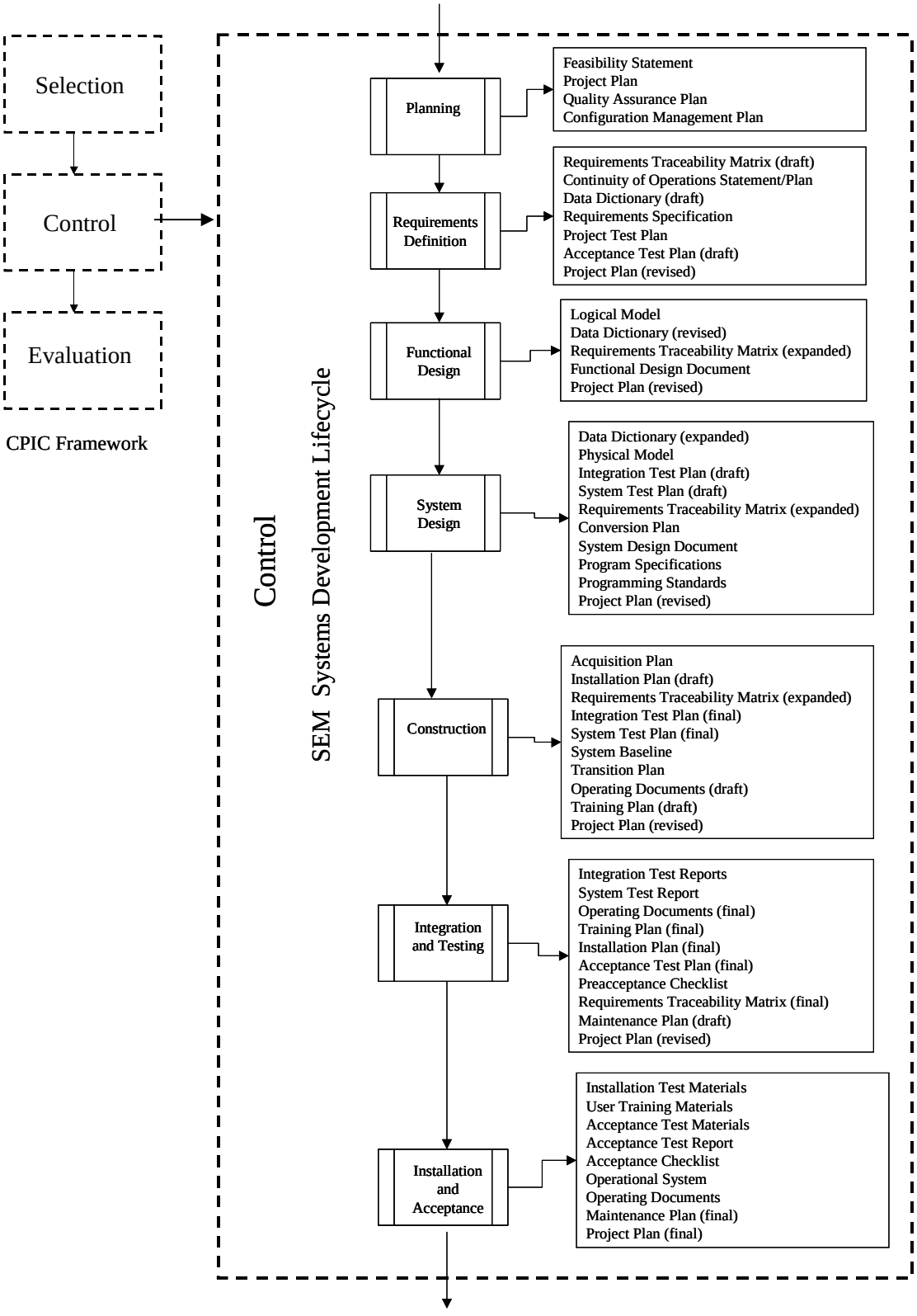
Resources: Additional information on these programs and others may be obtained by accessing the CIO's Web site.

Exhibit 1.1-1. Capital Planning and Investment Control Selection Phase & SEM



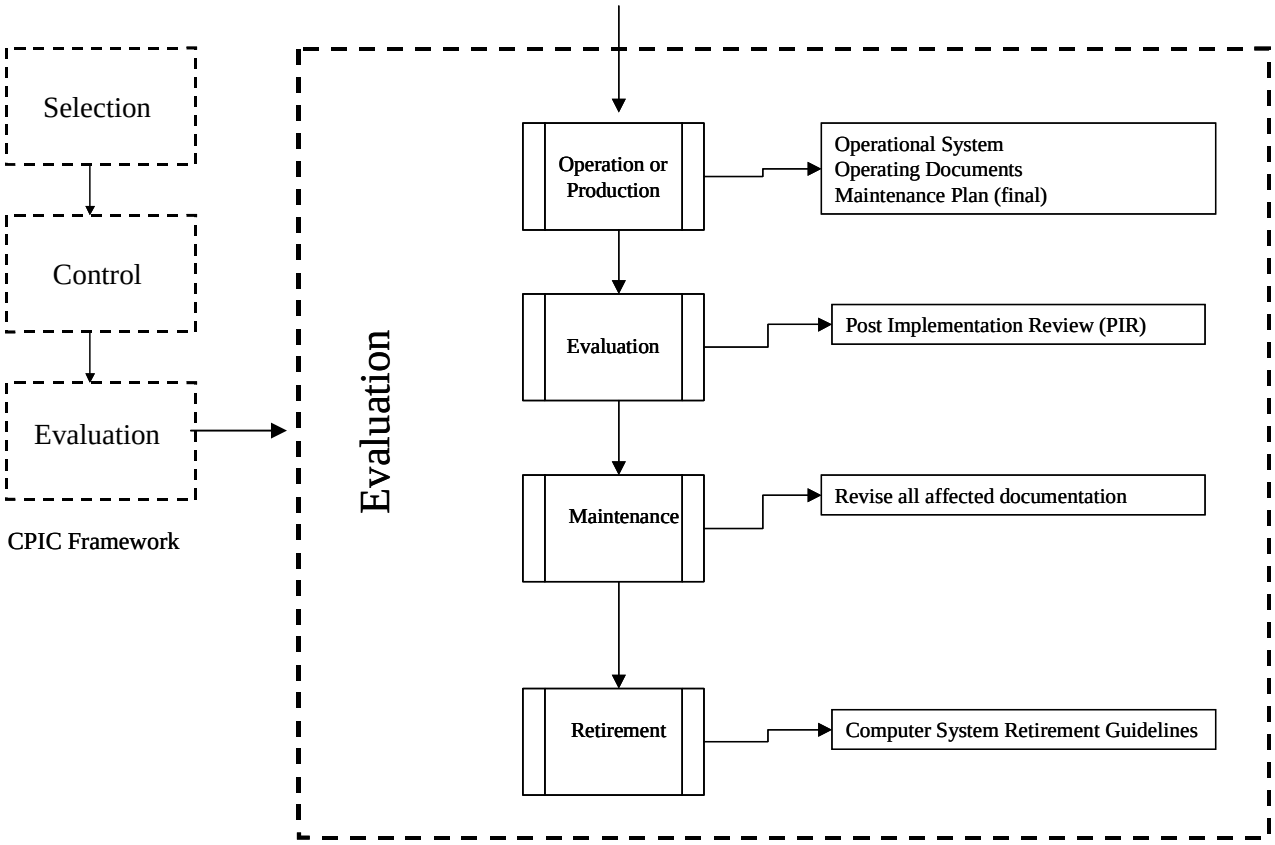
The SEM provides guidance to help meet the requirements of the Selection Phase in the areas of project benefits and costs (business case), project planning, user requirements, and acquisition planning.

Exhibit 1.1-2. Capital Planning and Investment Control Control Phase & SEM



The SEM provides comprehensive guidance to help meet the requirements of the Control Phase in the areas of requirements management, analysis of benefits and costs (business case), project planning, tracking, control, reporting, and assessment for, e.g., alignment to the Enterprise Architecture.

Exhibit 1.1-3. Capital Planning and Investment Control Evaluation Phase & SEM



The SEM provides guidance for planning system operation and maintenance, guidance for system retirement, and a comprehensive method for conducting system maintenance activities in the Evaluation Phase.

Section: **1.2 Organizational Implementation of Methodology**

Description: While the focus of the SEM is at a project level (see Section 1.3, Project Implementation of Methodology), it is recognized that there must be an organization-wide ability for managing information systems development, integration, and maintenance processes and quality oversight to ensure the delivery of high-quality products. Within this context, an organization is a DOE unit, (e.g., a Program Office or laboratory, within which, generally, many projects are managed.) The SEM integrates systems and infrastructure project management and quality assurance practices and is designed to be flexible. It can be adapted to accommodate the specific needs of any information systems engineering organization and all computing platforms used in the Department. When the SEM is adapted to be the organization's standard process for developing and maintaining systems, any additional specific or unique management processes should be integrated into the organization to help project managers and technical staff perform more effectively.

In a mature organization, the processes are institutionalized. They are documented, reusable, and consistent with the way the work is actually accomplished. The process definitions are updated when necessary, and improvements are applied when appropriate, with broad-based active involvement across the organization. Roles and responsibilities are clear and communicated throughout projects and across the organization. Organizational training ensures personnel are well trained so they can perform their roles effectively and efficiently.

The following tasks describe processes and activities complementary to those at the project level and aimed at maturing the entire organization in terms of capability to deliver high quality products.

Resource: Carnegie Mellon University, Software Engineering Institute, *Capability Maturity Model: Guidelines for Improving the Software Process*, Addison Wesley Longman, Inc., 1994

Tasks: The following tasks are involved in organizational implementation of the SEM.

1.2.1 Organizational Process Management
1.2.2 Organizational Training
1.2.3 Quality Oversight

Task: **1.2.1 Organizational Process Management**

Description: The goal of this task is to establish the organizational responsibility for lifecycle process activities that improve the organization's overall capability. The organization provides long-term commitments and resources to coordinate the development and maintenance of the process across current and future projects.

When the SEM is adopted by an organization within the Department, it becomes the organizational process for systems development and maintenance. The organization should then periodically assess the process and develop an action plan for improvement. Changes to the process are then communicated to those individuals within the organization responsible for implementing the process.

New processes, methods, and tools in limited use in the organization are monitored, evaluated, and, where appropriate, transferred to other parts of the organization. Three major components of Organizational process Management are Organizational Database, Organizational Library, and Communications.

Organizational Database
The organization manages and controls a database to collect and make available data on the systems process and resulting work products, (e.g., productivity data, quality measurements, and estimates of size, effort, and cost.) The database serves to improve project management planning and estimating by providing a resource for future system development efforts.

Organizational Library
An organizational library of systems process-related documentation is established and controlled. The library is cataloged for easy reference and the contents are made available for use by the project teams and other systems-related groups. The library contents are updated as appropriate.

Communications
A process is defined to ensure that the groups involved in implementing the systems engineering processes are informed of the organization's and projects' process development and improvement.

Work Products: An action plan is developed based on the periodic assessments. The action plan identifies guidelines for implementing the changes to address specified assessment findings and assigns responsibility for implementing changes.

An improvement plan is developed and maintained for process development and improvement activities. The plan uses the action plan and other improvement initiatives as primary input. The plan defines and schedules activities to be performed, assigns responsibility and identifies resources required for implementing the plan.

Work Products, continued: A database is established to function as a repository for organizational process and data (metrics) information. Members of the organization are trained in the use of, and have controlled access to, the database.

Review Process: Conduct structured walkthroughs for each of the written work products to remove as many defects as possible.

Task: **1.2.2 Organizational Training**

Description: An organization that is well prepared for the challenges posed by information systems engineering projects must ensure that its personnel are well trained to perform their roles effectively and efficiently. The goal of this task is to describe the areas of training that must be addressed to ensure the organization has a documented process in place to manage training activities on an ongoing basis.

The process should be based on documented organizational training standards. The standards should include how courses are to be developed (or standards that must be met where courses are procured) and how they are to be maintained according to these standards. Members of the training group (or vendors if training is acquired) need to have the necessary skills and knowledge to perform their training activities.

When determining the skills and knowledge needed for a project, the project teams are responsible for identifying their unique needs. Each project needs to evaluate its current and future skill needs and determine how these skills will be obtained. Some skills may be imparted through informal vehicles (e.g., on-the-job training, mentoring,) while other skills may need more formal training vehicles (e.g., classroom, self-study.) Appropriate vehicles need to be selected and used.

Responsibility for training needs to be identified and communicated. It may lie with a single manager within the organization, or may be shared by several managers, each responsible for one or more knowledge areas or subjects. The specific organizational responsibility for training needs to be identified, documented, and available for viewing by staff.

Waiver Process: A waiver procedure for required training needs to be established and used to determine whether staff already possess the knowledge and skills to perform their jobs.

Measurements: Measurements need to be identified, collected, and used to assess the status of training activities. Measurements should address areas such as the quality of the training, and if it meets the needs of the staff. Measurements and the organizational training should be reviewed with management on a regular basis.

Work Products: A written training policy describing how the organization will meet training requirements needs to be developed, communicated, and followed. The policy needs to be periodically reviewed and revised as appropriate based on feedback collected.

Work Products, A written training plan that addresses how the training needs of the organization will be met. The plan should include information such as how training needs will be identified, what training is required, how training will be delivered, the

continued: cost and resources required, organizational placement of the training function, who will be involved, when and how the plan will be reviewed and revised, and a work breakdown structure that identifies all of the activities involved.

Maintain records that training has been conducted and completed waivers, if and where appropriate.

Note: A written training plan is developed for each project (see *Section 3.8, Develop Project Plan*) and a training program is developed for system implementation and operation (*see section 7.9 Develop Training Program*).

Review Process: Conduct structured walkthroughs for each of the written work products to remove as many defects as possible.

Task: **1.2.3 Quality Oversight**

Description: The goal of this task is to establish the organizational responsibility for the quality oversight of information technology investments. While the lifecycle process activities for project implementation and maintenance are documented within the lifecycle stages of the SEM, an organizational quality oversight program provides long-term commitments and resources to coordinate the quality activities across current and future projects.

The quality oversight program should implement the appropriate level of management effort, and assume responsibility, accountability, and oversight for continued quality management process compliance within the organization. The quality oversight program should identify standards and best practices for product development, and ensure appropriate safety and security controls are in place, are effective, and reflect current accepted industry practices. The program should also ensure that project teams are aware of current DOE computer and cyber security directives and have coordinated the project with computer security staff.

Work Products: A written quality oversight program describing how the organization will ensure the development of high quality information technology investments needs to be developed, communicated, and followed. The program needs to be periodically reviewed and revised as appropriate, based on feedback collected.

The program should identify a point of contact for managing quality oversight, and ensuring project risk assessments are conducted to determine the appropriate level of quality assurance activities to be applied. The program should ensure the level of quality assurance is tailored to the site and project needs. The oversight program should oversee the development and implementation of quality assurance processes and procedures, and ensure the development and implementation of project quality assurance plans and production and delivery of quality products.

Review Process: Conduct a structured walkthrough of the quality oversight program to remove as many defects as possible.

Section: **1.3 Project Implementation of Methodology**

Description: This methodology integrates information systems engineering, project management, and quality assurance practices and is designed to be flexible. It can be adapted to accommodate the specific needs of any information systems project and all computing platforms used in the Department including standalone and networked mainframes, servers, desktops, and other computers.

Projects that were initiated prior to the awareness or usage of this document should plan to implement the methodology at the earliest feasible stage or the next release of the product. If a Project Plan already exists, make the revisions necessary to integrate the systems engineering, project management, and quality assurance practices, as appropriate. If a Project Plan does not exist, develop a plan that summarizes the activities and deliverables of the previous stages and incorporates the methodology activities and products into the subsequent stages.

The information systems engineering methodology presented here does not supersede, replace, or override more stringent requirements that may apply to specific projects such as scientific and technical practices, and security and safety issues associated with the Nuclear Weapons Complex.

Since the methodology cannot provide specific guidance for every possible situation, suggestions for adapting the methodology to accommodate projects of varying size, complexity, and criticality are provided in *Chapter 2.0, Lifecycle Model*. Samples of project plans for projects using the methodology are contained on the Software Quality & Systems Engineering web page.

Questions: If specific questions are generated concerning the interpretation or applicability of portions of the methodology, the project team should attempt to resolve them during the project review activities built into the stages of the lifecycle. The system owner/user(s) and other project stakeholders must concur with any adaptations that are made.

When questions about interpretation or applicability of the guidance to a specific project cannot be resolved by the project team, the issue should be submitted to the site authority for information systems engineering, such as the site Information Resources Management or Information Management organization, for advice or resolution. DOE Software Quality & Systems Engineering staff may also be consulted on the interpretation or applicability of the methodology via the Web page or Program Manger, who is located in the Office of the CIO.

Questions, continued:

Questions and issues will be analyzed by the site authority and a response made in one of the following ways.

- An immediate solution is determined and provided to the project team.
- The issue is submitted to personnel who are considered experts in the area in question. Once a solution is reached, it is provided to the project team.

Note:

It is important to also submit questions of interpretation or applicability and the site-specific resolution to the Departmentwide Software Quality & Systems Engineering Manager in the Office of the CIO. A change control board established by the Program Manager will determine if a modification to the *Systems Engineering Methodology* is needed to clarify processes or to provide additional adaptation suggestions. A central clearinghouse for all questions and resolutions will ensure that needed changes to the methodology are identified and implemented in a timely and consistent manner.

Section: **1.4 Submitting Change Requests**

Description: The Departmental information systems engineering environment is continuously changing as emerging technologies are integrated into projects, system owner/user requirements are expanded, and organizational needs evolve. The SEM will be revised, as needed, to reflect changes in the environment, improvements suggested through user feedback, and the maturation of information systems engineering capabilities.

Users of the methodology are encouraged to submit suggestions for improving its content and to report any practices that are difficult to understand or create an implementation problem for a project team.

Suggestions and problems should be submitted on the Change Request Form (*Exhibit 1.4-1*) that is provided at the end of this section or via e-mail from the Software Quality & Systems Engineering web site. If the form is not available or does not accommodate the type of request being made, submit a memo that describes the suggestion or problem.

The Change Request Form or memo should be submitted to the Departmentwide Software Quality & Systems Engineering Manager in the Office of the CIO located at Headquarters in Germantown, Maryland. The Change Request should be submitted through the site's information management organization. All requests will be evaluated and the originator of the request will be notified of the action taken.

Some requests will be handled immediately while others may require investigation by an ad hoc working group of knowledgeable personnel. In some cases, a request may not be appropriate for the current environment, but will be retained for future consideration.

Exhibit 1.4-1. SEM Change Request Form

SEM Change Request Form	
To be completed by Requestor	To be completed by SQ&SE Staff
Name:	Name:
Phone:	Phone:
Location:	Location:
Date:	Date Assigned:
Document Section:	Request Number:
Requested Change and Justification: Check if additional pages are attached ___	Change Classification Data (check one) Class I Change ___ Class II Change ___ Class III Change ___ Summary of Impact: Check if additional pages are attached ___
Change Impacts Section Number: _____ Pages: _____	Change Impacts Section Number: _____ Pages: _____
Working Group Actions	
Approval/Disapproval Reason(s): Date _____	
Additional Comments:	

Exhibit 1.4-2. SEM Change Request Instructions

SEM Change Request Instructions	
Requestor's Section	**Instructions**
Name: Phone: Location: Date:	Fill in your name, telephone number, location and date.
Document Section:	List the document section where you want to make a change.
Requested Change and Justification:	State the change that you want to incorporate and state your reasons for the change. Attach additional pages as needed.
Change Impacts:	List any section numbers and pages that will be affected by the proposed change.
**Send the completed Change Request Form to the DOE Software Quality & Systems Engineering Manager in the Office of the Departmental CIO, Germantown, Maryland.	
Analyst's Section	**Instructions**
Name: Phone: Location: Date Assigned:	Fill in your name, telephone number, location, and date you received the assignment.
Request Number:	Obtain the sequential number that will be used to track the request from the DOE Software Quality & Systems Engineering Office of the Departmental CIO.
Change Classification Data:	**Class I Changes** in policy, procedures, required actions, or deliverables are defined by Government units (Congress, Office of Management and Budget), or by DOE policies, procedures, and administrative requirements. These changes must be reviewed and approved by the DOE Software Quality & Systems Engineering Manager and incorporated into the methodology with the next update.
	Class II Changes in technology or development methodology are descriptions of innovations in the way information systems products are developed. These changes require review and concurrence by a working group and must be approved by the DOE Software Quality & Systems Engineering Manager. Changes in required deliverables may be implemented by the DOE Software Quality & Systems Engineering staff or recommended by a working group. The impact on the remainder of the methodology when such changes are incorporated require review and concurrence by a working group and approval by the DOE Software Quality & Systems Engineering Manager.
	Class III Changes in factual information (security requirements), wording, or corrections of typographical errors will be implemented as soon as possible to keep the methodology accurate and current. Corrections of typographical errors are implemented without review. All other changes require review and approval by the DOE Software Quality & Systems Engineering Manager.
Summary of Impact:	State what effect the proposed change would have on other sections of the document or the methodology.
Change Impacts:	List any section numbers and pages that will be affected by the proposed change.
Working Group	**Instructions**
Approval/Disapproval Reasons:	State whether the proposed change should be approved or disapproved. Give reasons for the decision. Indicate date of approval/disapproval.

Chapter: **2.0 Lifecycle Model**

Description: This chapter describes the lifecycle model used for the Departmental systems engineering methodology. This model partitions the information systems engineering lifecycle into eight major stages, as shown in *Exhibit 2.0-1, Information Systems Lifecycle Stages and Deliverables.* Each stage is divided into activities and tasks, and has a measurable end point (Stage Exit). The execution of all eight stages is based on the premise that the quality and success of the product depends on a feasible concept, comprehensive and participatory project planning, commitments to resources and schedules, complete and accurate requirements, a sound design, consistent and maintainable construction techniques, and a comprehensive testing program. The lifecycle stages and activities are described in the following chapters.

Intermediate work products are produced during the performance of the activities and tasks in each stage. These work products are inspected and can be used to assess system integrity, quality, and project status. As a result, adequacy of requirements, correctness of designs, and quality of the products become known early in the effort.

At least one time for each work product, a Structured Walkthrough is performed. A Structured Walkthrough is an organized procedure for reviewing and discussing the technical aspects of systems or software engineering work products including documentation. The walkthrough is usually conducted by a group of peers and may include reviewers outside the developer's immediate peer group. The *Structured Walkthrough Process Guide* provides detailed process information.

At least one time during each stage, an In-Stage Assessment is performed. An In-Stage assessment is an independent review of the work products and deliverables developed or revised during each lifecycle stage. The assessment is typically conducted by a Quality Assurance representative and the results are provided to the project manager. In-Stage Assessments are recommended after the achievement of all major project milestones and the completion of deliverable work products. The *In-Stage Assessment process Guide* provides detailed process information.

At the conclusion of each stage, a Stage Exit is initiated to review the work products of that stage and to determine whether to proceed to the next stage, continue work in the current stage, or abandon the project. The approval of the system owner and other project stakeholders at the conclusion of each stage enables both the system owner and the project manager to remain in control of the project throughout its life, and prevents the project from proceeding beyond authorized milestones. The *Stage Exit process Guide* provides detailed process information.

Description, continued:

The end products of the lifecycle are the information system product, the data managed by the system, associated technical documentation, and user training and support. The end products and services are maintained throughout the remainder of the lifecycle in accordance with documented configuration management procedures.

The lifecycle model provides a method for performing the individual activities and tasks within an overall project framework. The stages and activities are designed to follow each other in an integrated fashion, whether the stages of development are accomplished sequentially, concurrently, or cyclically. Project teams have the flexibility to adapt the lifecycle model to accommodate a particular development methodology (e.g., spiral development,) information systems engineering technique (e.g., prototyping and rapid application development,) or other project constraints.

The amount of project and system documentation required throughout the lifecycle depends on the size and scope of the project. System documentation needs to be at a level that allows for full system operability, usability, and maintainability. Typically, projects that require at least one work-year of effort should have a full complement of documentation. For projects that require less than one work-year of effort, the project manager and system owner should determine the documentation requirements. In addition, the project's security and quality assurance criteria may require the performance of other activities and the generation of additional documentation.

The requirements for documentation should not be interpreted as mandating formal, standalone, printed documents in all cases. Progressive documents that continuously revise and expand existing documentation, online documents, forms, reports, electronic mail messages, and handwritten notes (e.g., informal conference records) are some examples of alternative documentation formats. Project managers should verify documentation standards within their sites.

The following sections provide additional information about the lifecycle model.

Bibliography: The following materials were used in the preparation of the Lifecycle model chapter.

1. Booch, G., *Object-Oriented Analysis and Design,* 2nd edition, Benjamin Cummings, 1994.

2. Budd, T., *An Introduction to Object-Oriented Programming*, 2nd edition, Addison-Wesley, 1996.

3. Carnegie Mellon University, Software Engineering Institute, *Capability Maturity Model: Guidelines for Improving the Software Process,* Addison Wesley Longman, Inc., 1994.

4. Carney, D, & Oberndorf, P. "The Commandments of COTS: Still Searching for the Promised Land." Crosstalk 10, 5 (May 1997): 25-30.

5. Federal Acquisition Regulations. Washington, DC: General Services Administration, 1996.

6. Jacobson, I., *Object-Oriented Software Engineering,* Addison-Wesley, 1992.

7. Meyers, Craig & Oberndorf, Tricia. Open Systems: The Promises and the Pitfalls. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1997.

8. Open Systems Joint Task Force Baseline Study, 1996 [online].

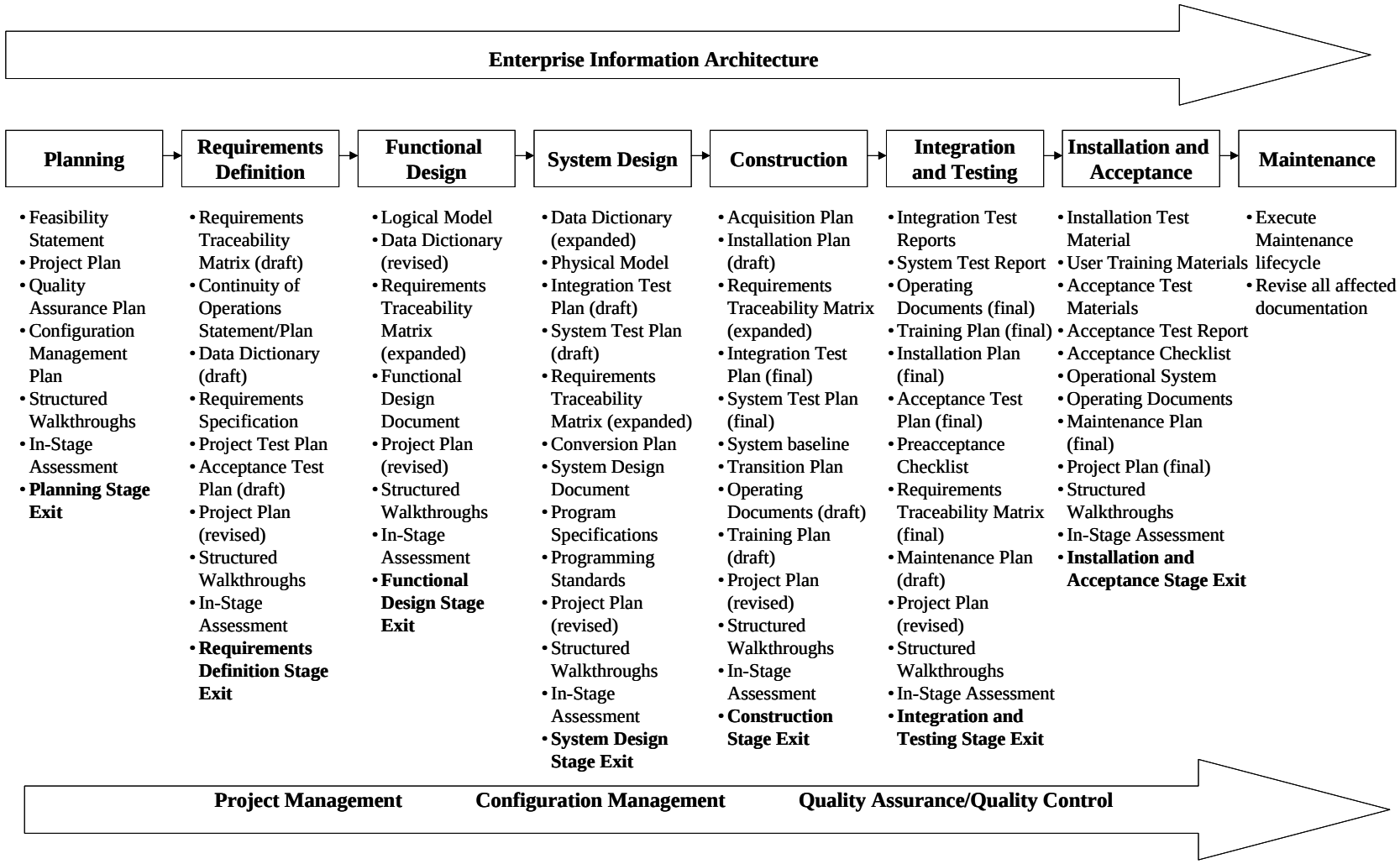
9. Open Systems Joint Task Force Case Study of U.S. Army Intelligence and Electronic Warfare Common Sensor (IEWCS), 1996 [online].

10. Pressman, Roger S., *Software Engineering - A Practitioner's Approach,* 4th edition, McGraw-Hill Companies, Inc., 1997.

11. Siy, Harvey, *Identifying the Mechanisms to Improve Code Inspections Costs and Benefits*, 1996 [online].

12. Yourdon, Edward, *Structured Walkthroughs*, second edition, YOURDON inc., New York, 1978.

Exhibit 2.0-1. Information Systems Lifecycle Stages and Deliverables



Note: For any given project, additional/other deliverables may be appropriate. All deviations should be documented in the project plan.

Section: **2.1 Project Sizes**

Description: The lifecycle model used in this information systems engineering methodology can be applied to projects of varying sizes. In this model, projects are divided into three sizes: large, medium, and small. Each project size uses the same lifecycle stages. Medium and small projects may compress or combine stages and required documentation in direct proportion to the size of the development effort. The major differences between project sizes are determined by the following items.

- The estimated total labor hours (the level of effort) required to complete the project.
- The use of cutting edge or existing technology.
- The type and extent of both user and system interface requirements.
- The project's contribution to, and impact on, the activities carried out by the system users and other Departmental organizations.

The requirements, constraints, and risks associated with the project also influence the determination of project size. The project size and any plans for adapting the lifecycle model are documented in the Project Plan, which is reviewed and approved by the system owner and other project stakeholders.

The following subsections provide descriptions of the three project sizes used in this lifecycle model. *Exhibit 2.1-1, Information Systems Project Sizes*, shows the level of effort and complexity measures used to define the three sizes.

Large Projects: Large information systems engineering projects are included in the system owner's organizational long-range plans. Departmentwide and site-specific projects are usually developed as large-sized projects and are likely to require a major acquisition of hardware and software. Typically, the larger the size and scope of the project, the greater the detail and coordination needed to manage the project. As risk factors and levels of effort increase, the scope of project management also increases and becomes a critical factor in the success of the project.

Medium Projects: Medium information systems engineering projects require less effort than large projects, typically use existing hardware and software, and might not be captured during the organizational long-range planning process. They are frequently developed to automate operations within a programmatic office or among a limited number of sites, and may be used to interface with other systems. Planning medium size projects within the context of the system owner organization's overall mission, and building in compatibility to the Departmental IT environment can improve the product's ability to interface with other users, organizations, and applications; and increase the product's longevity.

Small Projects: Small information systems engineering projects require minimal effort and use existing hardware and software. The operational details of a small project can easily be managed by the project manager, so formal documentation requirements are limited. A project is small when the system being developed will have limited functionality and use, meets a one-time requirement, or is developed using reusable code.

Exhibit 2.1-1. Information Systems Project Sizes

Complexity (and associated characteristics)	**Effort Required** (in staff months)		
	0-8	**9-24**	**25-n**
Low: - Existing or known technology - Simple interfaces - Requirements well known - Skills are available	Small	Small	Medium
Medium: - Some new technology - Multiple interfaces - Requirements not well known - Skills not readily available	Small	Medium	Large
High: - New technology - Numerous complex interfaces - Numerous resources required - Skills must be acquired	Medium	Large	Large

Note: Size is used as a guide to help determine the appropriate degree of project management, and whether any stages may be combined for a given effort. Within this context, size is a combination of level of effort required (all activities) and complexity of the requirements. Attributes of complexity include technology, team skills, interfaces, and level of understanding of requirements. Other factors that can influence adaptation include risk, visibility, and business impact.

Section: **2.2 Adapting the Lifecycle**

Description: The SEM implements well-defined processes in a lifecycle model that can be adapted to meet the specific requirements or constraints of any project. This section provides guidelines for adapting the lifecycle processes to fit the characteristics of the project. These guidelines help ensure that there is a common basis across all projects for planning, implementing, tracking, and assuring the quality of the work products.

The lifecycle model has built-in flexibility. All of the stages and activities can be adapted to any size and scope information systems engineering project. The lifecycle can be successfully applied to development projects, maintenance or enhancements, and customization of commercial software. The lifecycle is appropriate for all types of administrative, business, manufacturing, laboratory, scientific, and technical applications. For scientific and technical projects, adaptations to the lifecycle may be dictated by the project stakeholders or the requirements for reporting technical results in formal reports or journal articles.

Adaptations: The lifecycle can be compressed to satisfy the needs of a small project, expanded to include additional activities or work products for a large or complex project, or supplemented to accommodate additional requirements, (e.g., security requirements.) Any modifications to the lifecycle should be consistent with the established activities, documentation, and quality standards included in the methodology. Project teams are encouraged to adapt the lifecycle as long as the fundamental information systems engineering objectives are retained and quality is not compromised.

The following are some examples of lifecycle adaptations.

- Change the order in which lifecycle stages are performed.
- Schedule stages and activities in concurrent or sequential order.
- Repeat, merge, or eliminate stages, activities, or work products.
- Include additional activities, tasks, or work products in a stage.
- Change the sequence or implementation of lifecycle activities.
- Change the development schedule of the work products.
- Combine or expand activities and the timing of their execution.

Adaptations, continued:

The lifecycle forms the foundation for project planning, scheduling, risk management, and estimation. When a lifecycle stage, activity, or work product is adapted, the change must be identified, described, and justified in the Project Plan. The Project Plan is developed as a separate document and includes a description of the systems development lifecycle, which is the organization's standard process.

Exhibit 2.2-1, Adapting the Lifecycle, shows how stages can be combined to accommodate different size projects and information systems engineering techniques. *Notes* are provided throughout the lifecycle stage chapters to identify activities that have built-in project adaptation strategies. Adaptations should not introduce an unacceptable level of risk and require the approval of the system owner and other project stakeholders.

When adapting the lifecycle model, care must be taken to avoid the following pitfalls.

- Incomplete and inadequate project planning.
- Incomplete and inadequate definition of project objectives and requirements.
- Lack of a development methodology that is supported by information systems engineering preferred practices and tools.
- Insufficient time allocated to complete design before coding is started.
- Not defining and meeting criteria for completing one lifecycle stage before beginning the next.
- Compressing or eliminating testing activities to maintain an unrealistic schedule.

Sample Statements:

The following are sample statements that can be used in the Project Plan to describe different types of lifecycle adaptations. The first example shows a scenario where the Feasibility Study activity will not be conducted in the Planning Stage.

A Feasibility Study will not be performed for this project. The need for the product has been documented in several organizational reports and was

Sample Statements, continued:

included in the fiscal year long-range plans. The platform for the project is currently used for all applications owned by this organization. There are no known vendor packages that will satisfy the functional requirements described by the system owner.

The following is a sample statement that shows how work products from two different stages can be combined into one deliverable.

The Functional Design and System Design documents will be combined into one design document. A Stage Exit will be conducted when the design document is completed. To reduce the risk associated with combining the two documents, the project team will develop prototype screens and reports for review and approval by the system owner/user(s) as the prototypes are developed.

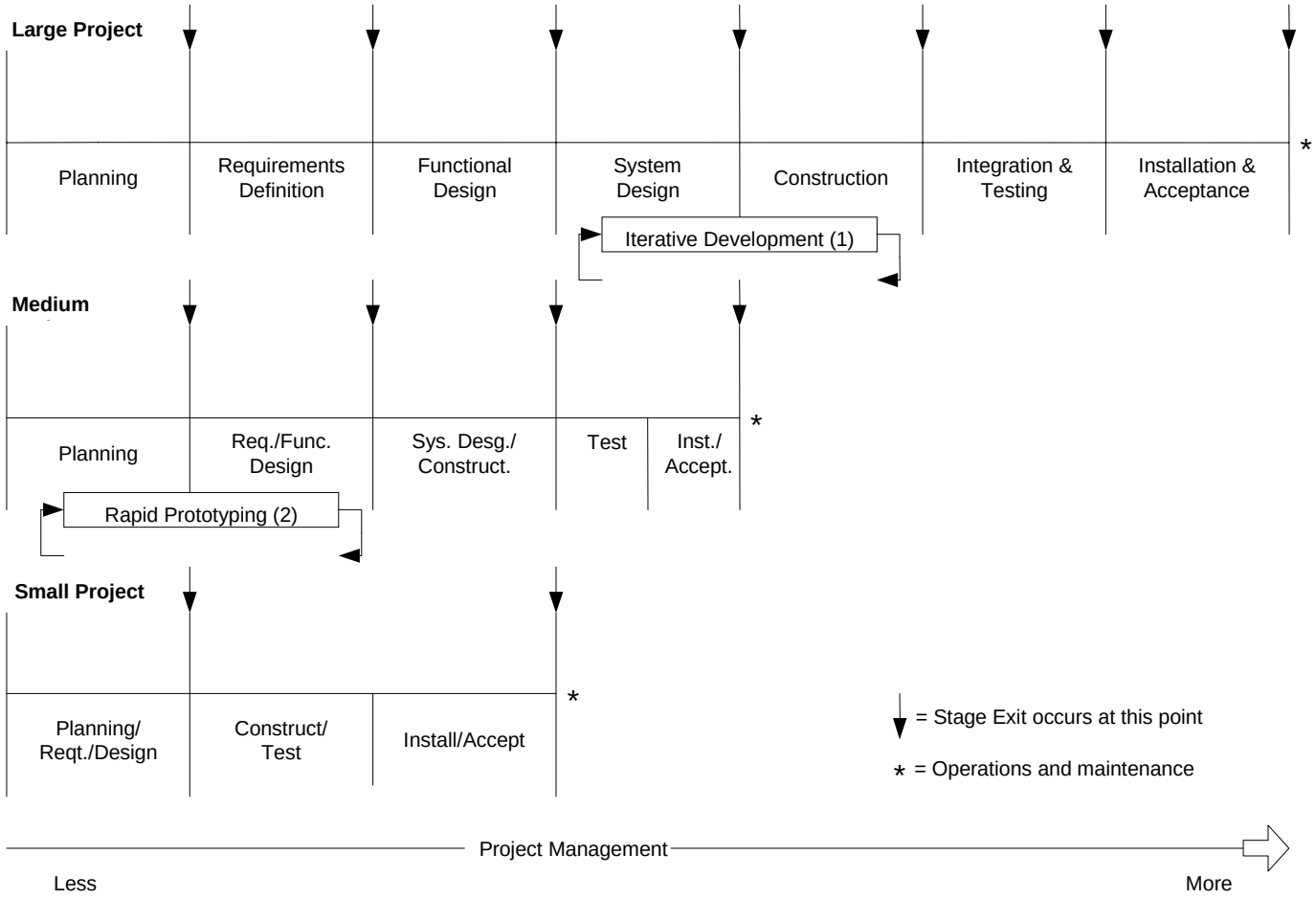
The following is a sample that shows how the eight lifecycle stages can be compressed into five stages for a small project.

This project will require 10 staff months of effort to enhance an existing application. The eight stages in the lifecycle will be combined into five stages as follows: (1) Planning, (2) Requirements and Design, (3) Construction and Testing, (4) Installation and Acceptance, and (5) Maintenance.

The following deviations will occur for document deliverables:

- *A Feasibility Study and an Analysis of Benefits and Costs will not be necessary due to the restricted software and hardware platform.*
- *The Requirements Specification will be limited to the statement of enhancement requirements.*
- *The Functional Design and System Design documents will be combined into one design document.*
- *An amendment package will be developed for the existing Users Manual.*

Exhibit 2.2-1. Adapting the Lifecycle



Note: Iterative development and rapid prototyping are optional techniques that can be used on any size project.

[1] Each iteration produces working function(s) from integrated program modules.
[2] May produce any or all of requirements, system architecture, functional design, system design.

Section: **2.3 Development Techniques**

Description: This section provides descriptions of some development techniques that can be used with the SEM. The descriptions include high-level instructions on how to adapt the lifecycle stages to accommodate the development technique. *Exhibit 2.2-1, Adapting the Lifecycle*, shows how some development techniques can be used with the information systems lifecycle model. The descriptions provided here are not intended to be a comprehensive list of development techniques.

Segmented Development: Segmented development is most often applied to large information systems engineering projects where the project requirements can be divided into functional segments. Each segment becomes a separate project and provides a useful subset of the total capabilities of the full product. This segmentation serves two purposes: to break a large development effort into manageable pieces for easier project management and control; and to provide intermediate work products that form the building blocks for the complete product.

The lifecycle processes and activities are applied to each segment. The overall system and software objectives are defined, the system architecture is selected for the overall project, and a Project Plan for development of the first segment is written and approved by the system owner.

Segments are delivered to the system owner for evaluation or actual operation. The results of the evaluation or operation are then used to refine the content of the next segment. The next segment provides additional capabilities. This process is repeated until the entire product has been developed. If significant problems are encountered with a segment, it may be necessary to reexamine and revise the project objectives, modify the system architecture, update the overall schedule, or change how the segments are divided.

Two major advantages of this approach are: the project manager can demonstrate concrete evidence that the final product will work as specified; and users will have access to, and use of, segments or functions prior to the delivery of the entire product.

Spiral Development: Spiral development repeats the planning, requirements, and functional design stages in a succession of cycles in which the project's objectives are clarified, alternatives are defined, risks and constraints are identified, and a prototype is constructed. The prototype is evaluated and the next cycle is planned.

Spiral Development, continued:

The project objectives, alternatives, constraints, and risks are refined based on this evaluation; then, an improved prototype is constructed. This process of refinement and prototyping is repeated as many times as necessary to provide an incrementally firm foundation on which to proceed with the project.

The lifecycle activities for the Planning, Requirements Definition, and Functional Design Stages are repeated in each cycle. Once the design is firm, the lifecycle stages for System Design, Construction, and Integration and Testing are followed to produce the final product.

Rapid Prototyping:

Rapid prototyping can be applied to any information systems development methodology (e.g., segmented, spiral.) Rapid prototyping is recommended for systems development that is based on a new technology or evolutionary requirements.

With the rapid prototyping technique, the most important and critical requirements are defined based on current knowledge and experience. A quick design addressing those requirements is prepared, and a prototype is coded and tested. The purpose of the prototype is to gain preliminary information about the total requirements and confidence in the correctness of the design approach. Characteristics needed in the final product, such as efficiency, maintainability, capacity, and adaptability might be ignored in the prototype.

The prototype is evaluated, preferably with extensive user participation, to refine the initial requirements and design. After confidence in the requirements and design approach is achieved, the final product is developed. The prototype might be discarded, or a portion of it used to develop the final product.

The normal documentation requirements are usually postponed with prototyping efforts. Typically, the project team, project stakeholders, and system owner agree that the prototype will be replaced with the actual product and required support documentation after proof of the model. The system that replaces the prototype should be developed using the lifecycle processes and activities.

Iterative Technique:

The iterative technique is normally used to develop products piece by piece. Once the system architecture and functional or conceptual design are defined and approved, system functionality can be divided into logically related pieces called "drivers."

Iterative Technique, continued:

In iterative fashion, the project team performs system design, code, unit test, and integration test activities for each driver, thereby delivering a working function of the product. These working functions or pieces of the product are designed to fit together as they are developed. This technique allows functions to be delivered incrementally for testing so that they can work in parallel with the project team. It also enables other functional areas, such as documentation and training, to begin performing their activities earlier and in a more parallel effort. In addition, the iterative technique enables progress to be visible earlier, and problems to be contained to a smaller scope.

With each iterative step of the development effort, the project team performs the lifecycle processes and activities.

Rapid Application Development:

Rapid Application Development (RAD) is a method for developing systems incrementally and delivering working pieces every 3 to 4 months, rather than waiting until the entire project is constructed before implementation. Over the years, many information technology projects failed because by the time the implementation took place, the business had changed.

RAD employs a variety of automated design and development tools, including Computer-Aided Software Engineering (CASE), advanced generation languages, visual development, and graphical user interface (GUI) builders, which get prototypes up and running quickly. RAD focuses on personnel management and user involvement as much as on technology.

Joint Application Development:

Joint Application Development (JAD) is a RAD concept that involves cooperation between the designer of a computer system and the end user to develop a system that meets the user's needs exactly. It complements other system analysis and design techniques by emphasizing participative development among system owners, users, designers, and builders. During JAD sessions for system design, the system designer will take on the role of facilitator for possibly several full-day workshops intended to address different design issues and deliverables.

Object-Oriented Development:

Object-oriented development focuses on the design of components that mimic the real world. A component that adequately mimics the real world is much more likely to be used and reused. The approach emphasizes how a system operates, as opposed to analysis, which is concerned with what a system is capable of doing. One of the most important advantages in using an object- oriented approach is the ability to reuse components. Traditional practices surrounding development often mitigate against reuse. Short-term goals are stressed because today's milestones must be achieved before any thought can be given to milestones that may be months or years away.

Object-Oriented Development, continued:

Borrowed or reused software code is often code that has already been tested, and in the end, may translate into cost savings. Object-oriented development may make code reuse much easier but, the amount of actual reuse may still depend on the motivation of the project managers, designers and developers involved. Code reuse can also lead to faster development. Object-oriented systems are easier to maintain because their structures are inherently decoupled. This usually leads to fewer side effects when changes have to be made. In addition, object-oriented systems may be easier to adapt and scale (i.e., large systems can be created by assembling reusable subsystems).

Typically, the object-oriented process follows an evolutionary spiral that starts with customer communication, where the problem is defined. The technical work associated with the process follows the iterative path of analysis, design, construction, and testing. The fundamental core concepts in object-oriented design involve the elements of classes, objects, and attributes. Understanding the definition and relationships of these elements is crucial in the application of object-oriented technologies.

It is recommend that the following object-oriented issues be well understood in order to form a knowledge base for the analysis, design, testing, and implementation of systems using object-oriented techniques.

- What are the basic concepts and principles that are applicable to object-oriented thinking?
- How should object-oriented projects be planned and managed?
- What is object-oriented analysis and how do its various models enable a systems engineer to understand classes, their relationships and behavior?
- What is a “use case” and how can it be applied to analyze the requirements of a system?
- How do conventional and object-oriented approaches differ?
- What are the components of an object-oriented design model?
- How are “patterns” used in the creation of an object-oriented design?
- What are the basic concepts and principles that are applicable for testing of object-oriented systems?
- How do testing strategies and test case design methods change when an object-oriented system is considered?

Object-Oriented Development, continued:

- What technical metrics are available for assessing the quality of object-oriented systems?

Work Product: The work products described in the SEM will be the same for many of the development techniques and it is the responsibility of the project manager to adapt the work products accordingly and document adaptations in the Project Plan.

Reference: Section 3.8, *Prepare Project Plan*, provides guidance for preparing a project plan.

Section: **2.4 Commercial-Off-The-Shelf (COTS) Products Based Projects**

Description: There is a current trend in information systems development to make greater use of Commercial-Off-The-Shelf (COTS) products, that is, to buy a ready-made system from a software manufacturer rather than developing it in-house from scratch. This carries with it a sense of getting a system that can do the job, at a reasonable cost, and getting new functions in subsequent releases over time. This practice is especially encouraged, and sometimes mandated, in government agencies. There can be many benefits in using COTS products including improving quality and performance, developing and delivering solutions more quickly, maintaining systems more cost effectively, and standardizing across the organization. The main characteristics of a COTS product are that it exists, is known to be proven, is available to the general public, and can be bought, leased, or licensed.

COTS and Open Systems: Many initiatives are under way in both private industry and government agencies, including DOE, to promote the use of an open systems approach, thereby anticipating even greater benefits than can be obtained from the use of COTS products alone. These initiatives are occurring because just buying COTS does not necessarily result in an "open" system. COTS products are not necessarily open, and they do not necessarily conform to any recognized interface standards. Therefore, it is possible that using a COTS product commits the user to proprietary interfaces and solutions that are not common with any other product, component, or system.

If the sole objective is the ability to capture new technology more cheaply, then the use of COTS products that are not open may satisfy requirements. However, considering that the average COTS component is upgraded every 6 to 12 months and new technology appears on the scene about every 18 to 24 months, any money that is saved by procuring a COTS product with proprietary interfaces may quickly be lost in maintenance as products and interfaces change.

In the midst of all this, interface standards provide a source of stability. Without such standards every change in the marketplace can impose an unanticipated and unpredictable change to systems that use products found in the marketplace.

COTS Planning Considerations: A COTS-based systems solution approach requires new and different investments including market research on available and emerging products and technologies, and COTS product evaluation and selection. The key to determining if the best solution is one which includes COTS products is to weigh the risks of straying from the three basic criteria - fully-defined, available to the public, and maintained according to group consensus - against what is to be gained over the

COTS Planning Considerations, continued:

long term. An open systems approach requires investments in the following areas early in a project's lifecycle and on an ongoing basis:

- Market surveys to determine the availability of standards
- Selection of appropriate applicable standards
- Selection of standards-compliant implementations

These costs/activities are the necessary foundation for creating systems that serve current needs and yet can grow and advance as technology advances and the marketplace changes. On an ongoing basis, it is important for project teams to stay informed in this area, with particular focus on:

- When revisions to specific standards are scheduled for release
- What changes are proposed in the new revision
- When ballots on the revisions are going to occur
- Where the implementations are headed

Skills Considerations:

The depth of understanding and technical and management skills required on a project team are not necessarily diminished or decreased because of the use of COTS or open systems. The skills and understanding needed increase because of the potential complexity of integration issues, the need to seriously consider longer-term system evolution as part of initial development, and the need to make informed decisions about which products and standards are best.

Types of COTS Solutions:

COTS products can be applied to a spectrum of system solutions, including (but not limited to) the following:

- Neatly packaged solutions such as Microsoft Office that require no integration with other components.

- COTS products that support the information management domain, such as Oracle or Sybase. These systems typically consist of both COTS products and customized components, with some "glue" code to enable them to work cooperatively.

- Systems comprised of a mix of COTS products and non-commercial products that provide large-scale functionality that is otherwise not available. Such systems typically require larger amounts of "glue" code to integrate the various components.

COTS Impact on the Project Lifecycle:

All systems engineering projects include planning, requirements definition, architecture definition, system design, code, test, and system integration activities. The use of COTS products has an impact on project lifecycle activities. The most fundamental change is that the system is now composed from building blocks that may or may not work cooperatively directly out of the box. The project team will require skilled engineering expertise to determine how to make a set of components work cooperatively - and at what cost.

This fundamental shift from development to composition causes numerous technical, organizational, management, and business changes. Some of these changes are obvious, whereas others are quite subtle.

Requirements Definition

For a COTS-based system, the specified requirements must be sufficiently flexible to accommodate a variety of available commercial products and their evolution. To write such requirements, the author should be sufficiently familiar with the commercial marketplace to describe functional features for which actual commercial products exist.

There is a critical relationship among technology and product selection, requirement specification, and architecture definition. If the architecture is defined to fulfill the requirements and then the COTS product is selected, there may be only a few or no available products that fit within the chosen architecture. Pragmatically, three essential elements--requirements, architecture, and product selection--must be worked in parallel with constant trade-offs among them.

Adaptation/Integration

Assembling COTS products presents new challenges. Although COTS products are attempting to simulate the "plug and play" capability of the hardware world, in reality, they seldom plug into anything easily. Most products require some amount of adaptation and integration to work harmoniously with other commercial or custom components in the system. The typical solution is to adapt each COTS product through the use of "wrappers," "bridges," or other "glueware." It is important to note that adaptation does not imply modification of the COTS product. Adaptation can be a complex activity that requires technical expertise at the detailed system and specific COTS component levels. Adaptation and integration must take into account the interactions among custom components, COTS products, any non-developmental item components, any legacy code, and the architecture including infrastructure and middleware elements.

COTS Impact on the Project Lifecycle, continued:

Testing

As the testing of COTS-based systems is considered, it must be determined what levels of testing are possible and needed. A COTS product is a "black box" and therefore changes the nature of testing. A system may use only a partial set of features of a given COTS product. In developing a test strategy and test plans, consideration should be given to issues such as should only the features used in the system be tested, and how does one test for failures in used features that may have abnormal behavior due to unknown dependencies between the used and unused features of a COTS product?

Maintenance

Maintenance also changes in very fundamental ways; it is no longer solely concerned with fixing existing functionality or incorporating new mission needs. Vendors update their COTS products on their schedules and at differing intervals. Also, a vendor may elect to eliminate, change, add, or combine features for a release. Updates to one COTS product, such as new file formats or naming convention changes, can have unforeseen consequences for other COTS products in the system. To further complicate maintenance, all COTS products will require continual attention to license expirations and changes. All of these events routinely occur. All of these activities may (and typically do) start well before an organization installs the system or a major upgrade. Pragmatically, the distinction between development and maintenance all but disappears.

Adapting the SEM for COTS Projects:

All systems engineering projects have a project lifecycle, require project management activities such as project planning, requirements definition, project tracking, configuration management, and quality assurance; and produce deliverables such as project plans, requirements specifications, configuration management plans, and test plans. At the same time, each project, whether COTS or traditional, can vary in scope, duration, technology used or operating platform. The SEM can be used as the project lifecycle for COTS-based projects as well as for traditional systems development and maintenance projects where all of the code is developed “in-house.”

The key to using the SEM effectively for COTS projects lies in adapting the lifecycle stages and deliverables to best suit the individual needs and characteristics of each particular project. See *Exhibit 2.4-1, Example of SEM Adapted for COTS Projects*, for an example of how to adapt the SEM for a COTS project. Stages should be combined as appropriate if, for example, a project will have a relatively small scope, and/or short duration, and/or will use known technology. On the other hand, the traditional number of stages may be

Adapting the SEM for COTS Projects, continued:

appropriate for large projects with new technology and long duration. *Exhibit 2.2-1, Adapting the Lifecycle*, shows how stages can be combined for all types of projects, based on the amount of project management required or anticipated.

Deliverables may be added to, or deleted from the standard list prescribed by the SEM (see *Exhibit 2.0-1, Information Systems Lifecycle Stages and Deliverables*). For COTS-based projects, the lifecycles stages will typically include "Evaluation," "selection," "customization," and "integration," and the project deliverables will typically include documents such as "Products to be Evaluated," and "COTS Solution Recommendations."

Documenting Deviations:

The adaptation (or deltas) from the standard SEM prescribed stages and deliverables are known as deviations. These deviations should be documented with an explanation in the project plan. Deviations from prescribed project deliverables should be documented with an explanation, and a statement, which describes how project risk is not elevated if a prescribed deliverable will not be produced.

Resources:

The following references are from the features section of the Carnegie Mellon University Software Engineering Institute Web site:

- Software Technology Review: COTS and Open Systems
- Monthly Features: The Opportunities and Complexities of Applying COTS
- Monthly Features: Discussion with Members of the SEI COTS-Based Systems Initiative
- Software Technology Review: Components-based Software Development/COTS integration

Exhibit 2.4-1. Example of SEM Adapted for COTS Projects

SEM Stages	SEM/COTS Project Stages	Deliverables	
		SEM/COTS Project Planned Deliverables	Adaptation vs. SEM Deliverables
Planning	Planning	$ Project Plan (includes WBS) $ Quality Assurance Plan	$ Prototype instead of Feasibility Statement $ CM Plan moved to Reqts. Definition
Requirements Definition	Requirements Definition	$ Functional Requirements Document $ Continuity of Operations Statement $ Products to be Evaluated $ Configuration Mgmt Plan $ Data Dictionary $ Traceability Matrix	$ The System and Acceptance Test Plans will be developed in the Evaluation and Selection Stage $ Acquisition Plan moved to Evaluation & Selection
Functional Design System Design	Evaluation and Selection	$ COTS Solution/Recommendation $ Acquisition Plan $ System Architecture $ System/Acceptance Test Plan $ Conversion Plan	$ Functional Design and System Design stages are combined $ System Architecture document replaces System Design document $ Logical Model, Physical Model, Construction Specifications, Coding Practices not applicable
Construction Integration & Testing	Customization, Integration and Testing	$ Solution Baseline $ Training Plan $ User Documentation $ System Maint. Documentation $ Transition Plan $ Security Plan $ System Test Report $ System Installation Plan $ Preacceptance Checklist	$ The Construction and Integration and Testing stages are combined $ Integration Test Plan not required
Installation and Acceptance	Installation and Acceptance	$ Acceptance Test Report $ User Training Materials $ Acceptance Checklist $ Operational System	$ No Deviations

Section: **2.5 Quality Reviews**

Description: This section describes the quality review and assurance mechanisms that are used with the SEM. The purpose of quality reviews is to assure that the established information systems development and project management processes and procedures are being followed effectively, and that exposures and risks to the current project plan are identified and addressed. The quality reviews facilitate the early detection of problems that could affect the reliability, maintainability, availability, integrity, safety, security, or usability of the software product. The quality reviews enhance the quality of the end work products and deliverables of a project.

Quality reviews are conducted as Peer Reviews, Structured Walkthroughs (SWAT), In-Stage Assessments (ISA), and Stage Exits. The quality review used depends on the work product being reviewed, the point of time within the project stage, and the purpose of the review.

Review Process: **Peer Review**

A peer review is an informal review of information systems engineering work products, including documentation, which can be conducted at any time at the discretion of the work product developer. These informal reviews are performed by the developer's "peers"-- frequently other developers working on the same project. Informal reviews can be held with relatively little preparation and follow up activity. Review comments are informally collected and the product developer determines which comments require future action. Some of the work products prepared are considered interim work products as they feed into a major deliverable or into another stage. Interim work products are ideal candidates for peer review; however, all work products benefit from peer reviews.

Review Process: **Fagan Inspection**

An alternative to the informal peer review and the SWT is the Fagan Inspection. Michael Fagan published an influential paper detailing a software inspection process used at IBM. Basically, it consists of six steps.

1. **Planning.** The artifact to be inspected is checked to see whether it meets certain entry criteria. If so, an inspection team, usually composed of up to four persons, is formed. Inspectors are often chosen from a pool of developers who are working on similar software or software that interfaces with the current artifact. The assumption is that inspectors familiar with the artifact will be more effective than those who are not.

2. **Overview.** The author meets with the inspection team. He or she provides background on the artifact, (e.g., its purpose and relationship to other artifacts.)

3. **Preparation.** The inspection team independently analyzes the artifact and any supporting documentation and records potential defects.

Review Process, continued:

4. **Inspection.** The inspection team meets to analyze the artifact with the sole objective of finding errors. The meeting is held on the assumption that a group of people working together finds defects that the members, working alone, would not.

   Before the meeting, one person is designated as the team leader or moderator, who orchestrates the meeting. Another person, designated as the reader, paraphrases the artifact. Defects are found during the reader's discourse and questions are pursued only to the point that defects are recognized. The issues found are noted in an inspection report and the author is required to resolve them. (Extensive solution hunting is discouraged during inspection.) The inspection meeting lasts no more than two hours to prevent exhaustion.

5. **Rework.** All issues noted in the inspection report are resolved by the author.

6. **Follow-up.** The resolution of each issue is verified by the moderator. The moderator then decides whether to reinspect the artifact depending on the quantity and quality of the rework.

Review Process: **Structured Walkthrough**

The structured walkthrough (SWT) is a more formal peer review and is prescribed by the SEM for all project deliverables. SWTs are used to find and remove errors from work products early and efficiently, and to develop a better understanding of defects that might be prevented. They are very effective in identifying design flaws, errors in analysis or requirements definition, and validating the accuracy and completeness of deliverable work products.

SWTs are conducted during all stages of the project lifecycle. They are used during the development of work products identified as deliverables for each stage (see *Exhibit 2.0-1*), such as requirements, specifications, design, code, test cases (scripts), and documentation. SWTs are used after the work products have been completed to verify the correctness and the quality of the finished product. They should be scheduled in the work breakdown structure developed for the project plan, where, in practice, they are sometimes referred to generically as reviews. SWTs should also be scheduled to review small, meaningful pieces of work. The progress made in each lifecycle stage should determine the frequency of the walkthroughs; however, they may be conducted multiple times on a work product to ensure that it is free of defects.

SWTs can be conducted at various times in the development process, in various formats, with various levels of formality, and with different types of participants. They typically require some advance planning activities, a formal procedure for

Review Process, continued:

collecting comments, specific roles and responsibilities for participants, and have prescribed follow-up action and reporting procedures. Frequently reviewers include people outside of the developer's immediate peer group.

Responsibility
Project Manager, Work Product Author, Reviewers

Work Products
A SWT Meeting Record is available for the reviewers to record errors found prior to the walkthrough session, and for the scribe to record information discussed during the walkthrough. Upon completion, the presenter or author of the work product compiles a SWT Management Summary Report and a copy is placed in the Project File.

Reference
The DOE guidance document titled *Conducting Structured Walkthroughs* provides a procedure and sample forms that can be used for SWTs. The document is available on the Software Quality & Systems Engineering web site.

Review Process:

In-Stage Assessment
The in-stage assessment (ISA) is a quality review that is conducted by a reviewer who is typically independent of the project. The reviewer assesses project processes, work products, and deliverables to verify adherence to standards and that sound information systems engineering and project management practices are being followed. This is particularly important when multiple deliverables are developed in a single lifecycle stage. The reviewer prepares an ISA report based on the information contained within the deliverables. An ISA does not require meetings among the involved parties to discuss the deliverable; however a meeting is often scheduled with the reviewer and the work product developer once the ISA report is completed in order to review the findings. Subject matter experts and documentation editors may be used in addition to the assessor to further improve the quality of work products.

An ISA can be conducted anytime during a stage whenever a deliverable is stable enough, or near the end of a stage to prepare for stage exit. An ISA can be conducted for each of the deliverables or one ISA for multiple deliverables depending on when the deliverables are made available for review and their size. ISAs are conducted in all stages of the project lifecycle and should be scheduled in the work breakdown structure developed for the project

Responsibility
Independent Reviewer, Project Manager

Review Process, continued:

Work Products

An ISA report form is prepared by the independent reviewer and is used to identify open issues that need to be resolved in this stage. The report is delivered to the project manager and a copy should be placed in the Project File.

Reference

The DOE guidance document titled *In-Stage Assessment Process Guide* provides a procedure and sample report form that can be used for in-stage assessments. The document is available on the Software Quality & Systems Engineering web site.

Review Process:

Stage Exit

The Stage Exit is a process for ensuring a project meets the DOE and project standards and milestones identified in the project plan. The Stage Exit is conducted by the project manager with the project stakeholders, (e.g., system owner and the following points of contact: user, quality assurance, security, architecture and standards, project manager's manager, and platform.) It is a high-level evaluation of all work products developed in a lifecycle stage. It is assumed that each deliverable has undergone several peer reviews and/or SWTs as appropriate and an ISA was conducted prior to the Stage Exit process. The Stage Exit focuses on the satisfaction of all requirements for the stage of the lifecycle, rather than the specific content of each deliverable.

The goal of a Stage Exit is to secure the concurrence (i.e., approval) of designated key individuals to continue with the project and to move forward into the next lifecycle stage. The concurrence is an approval (sign-off) of the deliverables for the current stage of development including the updated project plan. It indicates that all qualifications (issues and concerns) have been closed or have an acceptable plan for resolution. At a Stage Exit meeting, the project manager communicates the positions of the key personnel, along with qualifications raised during the stage exit process, issues that remain open from the ISA, and the action plan for resolution to the project team, stakeholders, and other interested meeting participants. The Stage Exit meeting is documented in summary form. Only one Stage Exit for each stage should be necessary to obtain approval assuming all deliverables have been accepted as identified in the project plan.

A Stage Exit is an effective project management tool that is recommended for all projects regardless of size. For small projects, stages can be combined and addressed during one Stage Exit.

Responsibility

Project Manager.

Review Process, continued:

Work Products

A summary of the Stage Exit meeting is prepared by the project manager or a designee and distributed to the meeting attendees. The summary identifies any issues and action items needed to obtain concurrence prior to proceeding to the next lifecycle stage.

Reference: The DOE guidance document titled *Stage Exit Process Guide* provides a procedure and sample report form that can be used for Stage Exits. The document is available on the Software Quality & systems Engineering web site.

Chapter: **3.0 Planning Stage**

Description: This is the first stage in the lifecycle of an information systems engineering project. In this stage, the system requirements are identified, the users' environment is analyzed, the project objectives and scope are defined, the high-level project and functional requirements are estimated, the feasibility of the project is determined, and the Project Plan, Quality Assurance Plan and Configuration Management Plan are developed and approved.

Project planning applies to all projects regardless of their size. Planning involves selecting the strategies, policies, programs, and procedures for achieving the objectives and goals of the project. Planning is deciding, in advance, what to do, how to do it, when to do it, where to do it, and who is going to do it.

The requirements identified in project related materials, (e.g., a business case document, are the primary input to the Project Plan.) The level of detail will vary depending on project size. The preparation of the Project Plan and related materials involves several critical planning issues such as the identification of preliminary requirements; staff, schedule, and cost estimates; the technical and managerial approaches that will be used; and the assessment of potential risks associated with the project. This information forms the foundation for all subsequent planning activities.

During this stage, the system owner and users are interviewed to: identify their business needs and expectations for the product; gain a common understanding of the task assignment; and determine how the project supports the DOE and organizational missions and long-range information resource management plans. The system owner is the organizational unit that is funding the project, and users are the DOE employees and contractors who will use the product.

In this stage, the project team should be focused on identifying what the project will automate, and whether developing an IT solution makes sense from business, cost, and technical perspectives. If the project is feasible, then time, cost, and resource estimates must be formulated for the project, and risk factors must be assessed. It is important for the project team to work closely with representatives from all functional areas that will be involved in providing resources, information, or support services for the project. The information gathered in this stage is used to plan and manage the project throughout its lifecycle.

This stage involves development of a Configuration Management Plan to track and control work products and a Quality Assurance Plan to assure the production

Description, continued:

and operation of high quality products on schedule, within budget, and within the constraints specified by the system owner and user.

Input:

The following items provide input to this stage.

- $ Requirements identified in project related materials, (e.g, a business case)
- $ Related project initiation materials

High-Level Activities:

The remainder of this chapter is divided into sections that describe specific high-level activities performed during this stage. These activities represent the minimum requirements for a large information systems engineering effort. *Notes* are provided, as applicable, to assist in customizing these lifecycle stage requirements to accommodate different size efforts. The high-level activities are presented in the sections listed below.

3.1 Project Management
3.2 Analyze User Environment
3.3 Define Project Objectives
3.4 Define Project Scope
3.5 Develop High-Level Project Requirements
3.6 Establish Communication With Functional Areas
3.7 Determine Project Feasibility
3.8 Develop Project Plan
3.9 Develop Quality Assurance Plan
3.10 Develop Configuration Management Plan

Output:

Several work products are developed during this stage. The work products listed below are the minimum requirements for a large project. Deviations in the content and delivery of these work products are determined by the size and complexity of a project. Explanations of the work products are provided under the applicable activities described in the remainder of this chapter.

- $ Project File
- $ Description of user environment
- $ Statement of project objectives
- $ Statement of project scope
- $ Statement of high-level project requirements
- $ Functional area contact list and project profile
- $ Summary of platform options

Output, continued:

- Statement of project feasibility
- Analysis of Benefits and Costs Report
- Feasibility Study Document
- Project Plan (or Software Development Plan)
- Quality Assurance Plan

$ Configuration Management Plan

A matrix showing the work products associated with each high-level activity is provided in *Exhibit 3.0-1, Planning Stage Activities and Work Products by Project Size*. The matrix also shows which work products are deliverables and whether they are required or optional for small, medium, and large projects.

Review Process: Quality reviews are necessary during this stage to validate the project and associated work products. The activities that are appropriate for quality reviews are identified in this chapter and chapter 2.0, Lifecycle Model. The time and resources needed to conduct the quality reviews should be reflected in the project resources, schedule, and work breakdown structure.

Peer Review
Requirements for a peer review are documented under *Review Process* at the end of each Task, Subtask, or Activity section in this stage. The DOE guide titled *Conducting Structured Walkthroughs* provides a procedure and sample forms that can be used for SWTs.

In-Stage Assessment
Schedule at least one ISA prior to the Planning Stage Exit process. Additional ISAs can be performed during the stage, as appropriate. An ISA is recommended after the completion of the Feasibility Study. The DOE guide titled *In-Stage Assessment Process Guide* provides a procedure and sample report form that can be used for ISAs.

Stage Exit
Schedule a Stage Exit as the last activity of the Planning Stage to enable the project approvers to review project deliverables and provide a concur/non-concur position to the project manager. The DOE guide titled *Stage Exit Process Guide* provides a procedure and sample report form that can be used for stage exits.

References: Chapter 2.0, Lifecycle Model, *Quality Reviews*, provides an overview of the Quality Reviews to be conducted on a project.

The three aforementioned DOE guide documents are available on the Software Quality & Systems Engineering web site.

Resource: DOE Software Quality & Systems Engineering web site.

Bibliography: The following materials were used in the preparation of the Planning Stage chapter.

1. Carnegie Mellon University, Software Engineering Institute, *Capability Maturity Model: Guidelines for Improving the Software Process*, Addison Wesley Longman, Inc., 1994.

2. Project Management Institute, *A Guide to the Project Management Body of Knowledge*, Pennsylvania, 1996.

3. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Lifecycle Processes*, IEEE Std 1074-1991, New York, 1992.

4. U.S. Department of Energy, *Analysis of Benefits and Costs (ABCs) Guideline: Volume 1, A Manager's Guide to Analysis of Benefits and Costs*, DOE/MA-0342, June 1988.

5. U.S. Department of Energy, *Analysis of Benefits and Costs (ABCs) Guideline: Volume 2, An Analyst's Handbook for Analysis of Benefits and Costs*, DOE/MA-0343, June 1988.

6. U.S. Department of Energy, *Departmental Information Systems Engineering: Volume 1, Information Systems Engineering Lifecycle*, September 2000.

7. U.S. Department of Energy, *DOE/NV Software Management Plan*, Nevada Operations Office, May 1991.

8. U.S. Department of Energy, *Software Management Guide*, DOE/AD-0028, 1992.

Exhibit 3.0-1. Planning Stage Activities and Work Products by Project Size

Work Activity		Project Size L	M	S	Work Product	Deliverables L	M	S
3.1	Project Management	R	R	R	Project File	N	N	N
3.2	Analyze User Environment	R	R	R	Description of User Environment	I	I	I
3.3	Define Project Objectives	R	R	R	Statement of Project Objectives	I	I	I
3.4	Define Project Scope	R	R	R	Statement of Project Scope	I	I	I
3.5	Develop High-Level Project Requirements	R	R	R	Statement of High-Level Requirements	I	I	I
3.6	Establish Communication With Functional Areas	R	R	A	Functional Area Contact List and Project Profile	N	N	N
3.7	Determine Project Feasibility	R	R	R	Summary of Platform Options Risk Assessment Record of Feasibility Review Meeting Statement of Feasibility Analysis of Benefits and Costs Report Feasibility Study Document	I I N R A A	I I N R A A	A I N A A A
3.8	Develop Project Plan	R	R	R	Project Plan	R	R	R
3.9	Develop Quality Assurance Plan	R	R	R	Quality Assurance Plan	R	R	R
3.10	Develop Configuration Management Plan	R	R	R[2]	Configuration Management Plan	R	R	R
2.5	Conduct Structured Walkthroughs	R	R	A	Structured Walkthrough Management Summary Report	N	N	N
2.5	Conduct In-Stage Assessment	R	R	A	ISA Report Form[1]	N	N	N
2.5	Conduct Planning Stage Exit	R	R	A	Stage Exit Meeting Summary	N	N	N

Size: L = Large, M = Medium, S = Small

Minimum Requirements: R = Required, A = As Appropriate, N = Not a Deliverable[3], I = Input to other Deliverable

O = Optional work product

[1] = Completed by reviewer

[2] = Can adapt an existing plan

[3]A deliverable is a work product that is identified in the Project Plan and is formally delivered to the system owner and other project stakeholders for review and approval.

Activity: **3.1 Project Management**

Responsibility: Project Manager

Description: Project management is the process of applying knowledge, skills, tools and techniques to activities in order to satisfy stakeholder needs and expectations of a project by completing objectives on time, within budget, and according to specifications. The project management processes have been defined within five major groups; the initiating processes, the planning processes, the executing processes, the controlling processes and the closing processes. The project management processes are interrelated and activities and tasks are interwoven in the information systems engineering methodology stages.

The initiating processes are executed when the project manager and team recognize that a project or phase should begin and commit to do so. The planning processes are executed when the project manager devises and maintains a workable plan to accomplish the business need that the project was undertaken to address. The executing processes are accomplished when the project manager coordinates people and other resources to carry out the plan. The controlling processes are completed when the project manager ensures that the project objectives are met by monitoring and measuring progress and takes corrective action when necessary. The closing processes are accomplished when a formal acceptance of the project or phase brings the project or phase to an orderly end.

The Planning Stage identifies the methodology that will be used on DOE projects and how the project management processes are implemented. The project manager works with the project team to establish the goals and expectations for the project.

Work Product: A substantial amount of information that may be useful in later stages in the information systems engineering process is gathered during the Planning Stage. Create a centrally maintained Project File that can be used as the repository for all project information gathered during the Planning Stage and for all work products developed throughout the project lifecycle. The Project Manager should verify that all pertinent project information and documentation are placed in the Project File on a timely basis.

Resource: The Software Quality & Systems Engineering web site provides project management tools and techniques.

Tasks: The following tasks are involved in Project Management.

3.1.1 Subcontractor Management
3.1.2 Risk Management
3.1.3 Performance Measures
3.1.4 Enterprise Architecture

Task: **3.1.1 Subcontractor Management (as appropriate)**

Description: The purpose of subcontractor management is to select qualified subcontractors and manage them effectively. A subcontractor is an individual, partnership, corporation or association that contracts with an organization (i.e., the prime contractor) to design and/or develop the system or one or more system components. The following guidance provides generally accepted industry practices for subcontractor management and should not be construed as an interpretation of a DOE contract. These practices keep the doors of communication open and are conducive to the delivery of the expected product.

When a project being managed by the prime contractor requires a subcontractor, the project manager is responsible for ensuring that the teaming partners and subcontractors are held to the same standards and guidance as the prime contractor. The prime contractor and the subcontractor should agree and document their commitments and maintain ongoing communications throughout the project.

The work to be subcontracted should be defined and planned. Follow any Departmental, site-specific, or contractor processes for documenting and planning the agreement. When selecting the subcontractor, the evaluation should be based on the subcontract bidders' ability to perform the work, according to the Departmental, site-specific, or contractor evaluation and selection processes.

The contractual agreement between the prime contractor and the subcontractor is used as the basis for managing the subcontract. In some cases a subcontractor may be already tasked to DOE and work products for a specific project may not require a separate contractual agreement. In that case, the responsibility of the subcontractor should be documented in the Project Plan and Work Breakdown Structure and used as the basis for managing the subcontract.

If the subcontractor is not following the SEM, a subcontractor systems development plan which documents the methodology to be used should be reviewed and approved by the prime contractor and the DOE Program Manager. Once approved, the subcontractor's systems development plan is used for tracking the activities and communicating status. Changes to the statement of work, subcontractor terms and conditions and other commitments are resolved according to applicable Departmental, site-specific, or contractor processes.

Description, continued:

The prime contractor is ultimately responsible for the subcontractor's work. Therefore, the prime contractor's project manager should conduct quality reviews of the work products of the subcontractor or ensure that the subcontractor does so. The prime contractor's quality assurance representative should monitor the subcontractor's quality assurance activities as documented per Departmental or site-specific processes.

The prime contractor's configuration management group should monitor the subcontractor's activities for configuration management per the project's systems configuration management plan.

The prime contractor should conduct acceptance testing as part of the delivery of the subcontractor's products or ensure that the subcontractor does.

Periodic technical reviews and interchanges are held with the subcontractor. The subcontractor's performance is evaluated on a periodic basis and the evaluation is reviewed with the subcontractor. The prime contractor's management should conduct periodic status/coordination reviews with the subcontractor's management.

Task: **3.1.2 Risk Management**

Description: Risk is the possibility of loss. It is a function of both the probability of an adverse event occurring and its impact. The impact can manifest itself in a combination of financial loss, time delay, and loss of performance. A risk is the precursor to a problem, (i.e., the probability that, at any given point in the project lifecycle, the predicted goals cannot be achieved with available resources.) Risk cannot be eliminated but it can be managed. Risk management is critical to the success of any effort and is a strategic aspect of all projects.

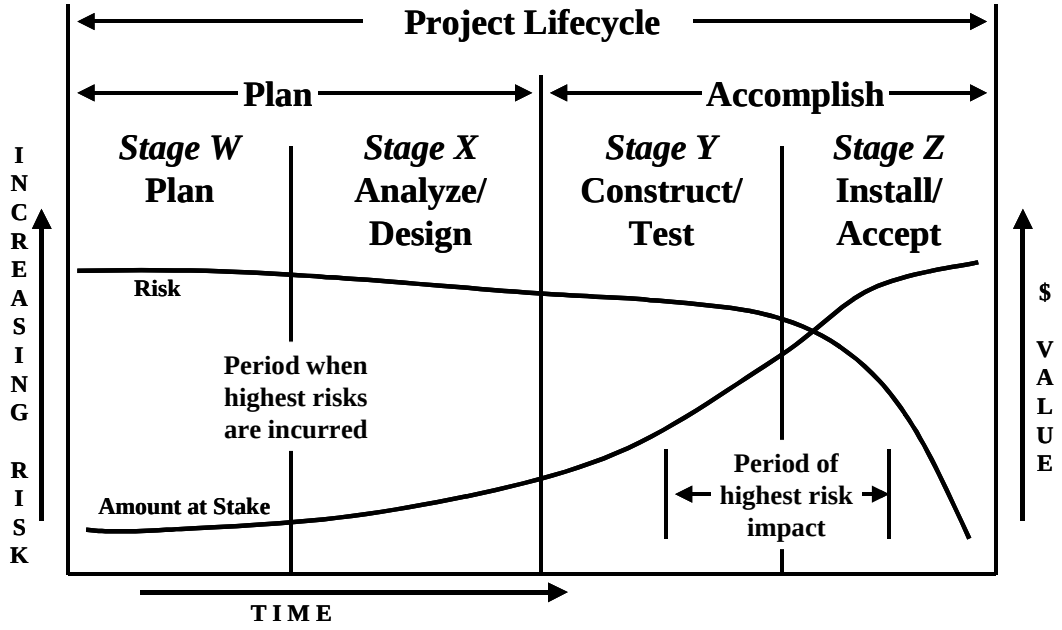


Exhibit 3.1-1. Typical Lifecycle Profile – Risk vs. Amount at Stake

Risk and the amount at stake will vary according to the project stage. As depicted on *Exhibit 3.1-1, Typical Lifecycle Profile - Risk vs. Amount at Stake*, risk generally remains relatively high during project planning and product design, but, because of the relatively low level of investment to this point, the amount at stake remains low. In contrast, during product construction, installation, and acceptance the level of risk progressively decreases as there are fewer and fewer unknowns. At the same time, the amount at stake steadily rises as the necessary resources are progressively invested to complete the project. The figure also shows that the period of highest risk impact occurs during the last two stages of the project lifecycle.

Description, continued:

All projects need to be subjected to a degree of assessment to determine the level of risk associated with the project and provide risk-related input to project decision-making. All corporate and major/large projects should undergo a formal risk assessment, and develop appropriate plans to mitigate and contain risks. Other projects may undertake a less rigorous and formal approach and may include a risk statement in the project plan rather than developing separate plans.

Project Risk Management:

Project risk management is the process of identifying, analyzing and responding to project risks that could result in cost and schedule overruns, and/or project failure. Risk management begins with an assessment of the environmental, operational and technical risks prior to the establishment of a project.

Once the project has been approved and is initiated, risk management becomes an integral part of the project. Risk assessments should be conducted at logical checkpoints, or when key decisions are being made throughout the project. Risk assessments help assure that positive events are maximized and that adverse events are minimized, (i.e., that the response to the risk assures that the risk is avoided, mitigated or accepted.) The elements of risk management include identification, quantification, and response development and control.

Risk Identification:

Risk identification consists of determining which risks, both internal and external, are likely to affect the project and documenting the characteristics of each risk. Products that involve proven technology will involve less risk and cause less cost and schedule impact than those that require innovation or invention. Projects that are mandated by law are less impacted by high-risk assessments because funding is generally made available and not likely to be eliminated. If a high level of risk is evident at a project's inception, and no tradeoff exists in the form of proven and affordable technology, a decision must be made as to whether the project is realistic or economically feasible.

Common influences for risk include changes in requirements, unrealistic requirements, changing technologies, unrealistic schedules, fluctuating or reduced budgets, ill defined or understood roles and responsibilities, inadequate estimates, insufficient skilled staff and low morale. The WBS, cost and duration estimates, staffing plan, and historical data will also be reviewed to identify possible risks.

Risk Quantification:

Risk quantification is the process of evaluating risks and risk interactions to assess the range of possible project outcomes. It is primarily concerned with determining which risk events warrant a response. It is complicated by opportunities and threats that can interact in unanticipated ways (schedule delays), single risks that can cause multiple effects, and conflicting stakeholder opportunities, just to name a few.

Risk Response Development: Risk response development is the definition of methods or steps that can be taken to reduce or eliminate risk. One method is avoidance of a specific risk, which can be accomplished by eliminating the cause. For example, goods or services can be acquired from outside the immediate project organization, such as subcontracting to a firm that has the experience with a particular technology. Another way is to develop a contingency plan, which defines actions to be taken if an identified risk event should occur. A third way is to hold subcontractors to the same project specifications and quality assurance standards as the prime contractor.

Contingency plans can reduce (mitigate) the expected monetary value of risk by reducing the probability of occurrence. The consequences of risk can be accepted and alternative strategies for changing the planned approach can be developed. Proceeding with a project with high risk must be documented and agreed to by the customer and stakeholders.

Risk Response Control: Risk control involves executing the risk management plan to respond to risk events over the course of the project. When changes occur, the basic cycle of identifying, quantifying, and responding is repeated. Even the most thorough and comprehensive analysis cannot identify all risks and probabilities, therefore, control and iteration of the risk assessment process are required.

Work Product: Work products from this task may consist of risk mitigation strategies, a risk management plan (separate document or part of project plan), and contingency plans. The information contained in these work products should be considered when making project decisions regarding, (e.g., schedule, resources, budget.)

Review Process: A peer review or structured walkthrough should be conducted on any work product produced.

Resources: Risk assessment tools are available from the Software Quality & Systems Engineering web site. Risk assessment guidance is available from the Software Quality Assurance Subcommittee (SQAS) Web site.

Task: **3.1.3 Performance Measures**

Description: The purpose of measuring performance is to enable the project team to collect and report targeted project data to track and assess project progress, product quality, project success, and customer satisfaction. Fundamentally, performance measurement is the ongoing monitoring and reporting of project accomplishments, particularly progress towards pre-established goals. Performance measures may address the type or level of project activities conducted (process), the direct products and services delivered by a project (outputs), and/or the results of those products and services (outcomes).

An effective set of performance measures will yield information, on a focused set of metrics, to provide a balanced view of project performance that can be used to make decisions to improve the project management process. Performance measures also provide for accountability by defining what is expected, and when it is expected, and can trigger action to be taken if planned achievements do not occur.

Requirement: In the typical sequence of performance measurements related events for an information systems engineering project, performance measures are identified and documented, measurement data is collected, metrics are developed, and indicators are obtained. As part of the planning process for each project, performance measures and management processes that monitor actual performance against expected results should be clearly established and documented. Measurements can include resource and cost goals, schedule and progress goals, trade-offs and risk outcomes, product quality goals, and customer satisfaction goals.

Basic Measurement Categories: *Measures of efforts.* Efforts are the amount of resources, in terms of money, people, and materials applied to a program or project.

Measures of accomplishments. Accomplishments are milestones achieved with the resources used. There are two types of accomplishments - outputs and outcomes. Outputs relate to the quantity of goods or services produced; outcomes relate to the results of providing those outputs.

Measures that relate efforts to accomplishments. These measures are associated with resources or cost relative to accomplishments achieved. They provide information about the production of an output at a given level of resource use and demonstrate an entity's capability when compared with previous results, internally established goals and objectives, generally accepted norms or standards, or results achieved by similar entities.

Types of Performance Measures:

Process Metrics	Increase capability level (i.e., SEI-CMM levels) Do more with less (shorter schedule, less resources) Improve quality (less defects, less re-work)
Project Metrics	Track project progress Assess project status Award contract fees
Product Metrics	Determine product quality Identify defect rates Ensure product performance

Typical Metric Categories:

Schedule	Actual vs. planned: Schedule and progress
Budget	Actual vs. planned: Resources and cost
Functionality	Delivered vs. planned: Product characteristics, Technology effectiveness, Process performance, Customer satisfaction

Measures vs. Indicators: Example: Finding defects in products (e.g. a Requirements Document)

Basic Measures: - No. of requirements reviewed - No. of reviewers involved - No. of defects found - Effort expended	Indicators: Efficiency - No. reviewed/effort - No. reviewed/time - No. found per effort, time Indicators: Effectiveness: - % found of those expected

Performance Measure Examples: A comprehensive set of performance measures examples that are typical for many information systems engineering projects are provided on the Software Quality & Systems Engineering web site.

Work Product: Work products from this task include identification of specific performance measurements for the project, when the measurement data will be collected, what means will be used to collect the data, what will be reported, and to whom the report(s) will be distributed. This information may be included in the project plan or provided as a separate document.

Review Process: A peer review or structured walkthrough should be conducted on any work products produced.

Resources: A White Paper and additional information on metrics and performance measurements is available from the SQ&SE web site.

Task: **3.1.4 Enterprise Architecture**

Description: The DOE Enterprise Architecture (EA) continues to evolve towards the intended target state. The Department recognizes that an EA strengthens management of its information and technology resources by:

- Supporting unified capital planning and investment control for information technology (IT).
- Improving the governance and effectiveness of the Department's IT planning and budgeting processes.
- Increasing the efficiency, effectiveness, interoperability, and standardization of its major information systems.
- Establishing business, data, applications, and technology infrastructures at the Departmental level.
- Providing a migration strategy to a technology environment that meets the Department's strategic business objectives.

In addition to this internal focus, DOE recognizes the need to integrate external policy directions as defined by Congress and the Administration into its IT initiatives, including cyber and other security initiatives. The Office of Management and Budget (OMB) stresses the importance of a robust capital planning and investment control mechanism linked to an equally robust EA.

Requirement: For each DOE information systems project, a requirement for the project's architecture to be in alignment and compliance with the DOE target EA should be included in the project's Requirements Specifications. This, like all other project requirements, should be solutioned and tracked through to implementation, and should be included in the traceability matrix to ensure an audit trail and closure.

EA Layers: DOE is using the Chief Information Officer's Council, *Federal Enterprise Architecture Framework (FEAF)*, Version 1.0, September 1999, to structure its EA. The FEAF is one of the three architecture frameworks recommended by *A Practical Guide to Federal Enterprise Architecture*, Version 1.0, February 2001. The graphic below, *Exhibit 3.1-2 DOE EA Framework*, portrays the key components of the DOE Enterprise Architecture, as follows.

$ A business layer describing current business lines and functions and delineating a future business environment based on Departmental business objectives and long-term strategies.

$ A data layer summarizing existing information resources and establishing a target data environment.

EA Layers, continued:

- An application layer characterizing existing computer applications and defining a set of future applications to better meet DOE information processing and management requirements.

$ A technology layer depicting existing computer hardware, software, and communications resources and outlining a future technology environment.

$ A cross-cutting security layer summarizing current business, data, application, and technology security considerations and mitigation measures and establishing the components of a strengthened and technically up-to-date security infrastructure.

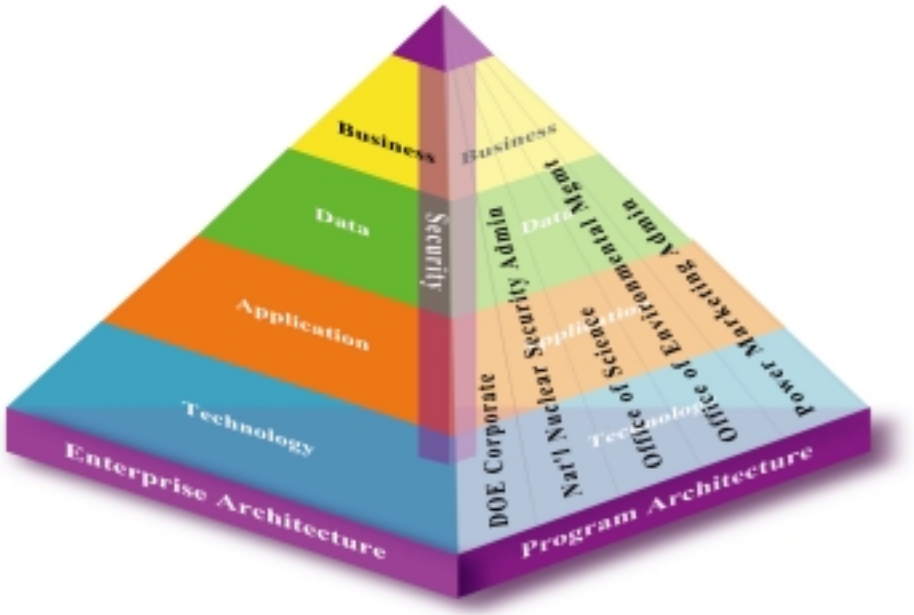


Exhibit 3.1-2. DOE EA Framework

EA Framework: The EA establishes an analytical framework by defining linkages among the business, data, application, and technology components of the architecture. The baseline (as is) and target (to be) architecture layers and the analytical framework establish the basis for developing a transition plan that defines the business process, data management, applications development, and infrastructure projects necessary to transition DOE to the target environment.

The figure above also indicates another important aspect of the Department's EA - its relationship with Program architectures. DOE Program areas are developing their architectures to meet important programmatic business objectives. The DOE EA will not replace these architectures but will serve to frame them from a Departmental perspective. The DOE EA will be enriched by the contents of the Programmatic architectures. A Program architecture can be considered as a *vertical sliver* that represents an in-depth architecture view of one part of the DOE enterprise.

EA Components: The DOE EA contains the following components.

- Architecture Drivers. Identifies the external stimuli that cause the EA to change. Business drivers, such as the President's Management Agenda, redefine core Federal business needs. Design drivers represent new ways of meeting Federal business needs, such as e-Government initiatives.

- Strategic Direction. Guides the development of the target architecture by incorporating a vision and strategic direction for the DOE enterprise.

- Baseline Architecture. Represents the current state of the DOE enterprise. The current business architecture defines the existing business lines and functions. The current design architecture defines the implemented data and applications, and the technologies used to support the business needs.

- Target Architecture. Represents the target state for the enterprise based on strategic direction. The target business architecture defines the enterprise's future business environment. The target design architecture defines the future data, applications, and technology that support business needs.

- Transitional Processes. Identifies the business processes, data management, application development, and infrastructure projects necessary to move the enterprise from the current to the target architecture in compliance with the architecture standards.

- Segments. Focuses on a subset of the DOE enterprise to structure initiatives with particular importance or impact such as the "Quicksilver" or Innovative Department of Energy e-Government Applications (IDEA) initiatives.

- Other Architectures. Provides documentation on program architectures.

- Standards**.** Describes the technical reference model and related IT standards relevant to the current architecture and evolutionary paths to keep pace with technology trends.

EA Evolution: The DOE EA will continue to evolve and expand. It will evolve to meet additional OMB architecture requirements such as providing a workforce analysis. A workforce analysis indicates the impact of the target environment on the existing Federal workforce and articulates strategies for improving or changing workforce composition and skills. The DOE EA will also evolve as additional or new information becomes available particularly in relation to DOE applications and technology resources. The evolution of the DOE EA is critical to ensuring that it can effectively serve DOE decision-makers as they consider and decide on IT strategies and investments.

Activity: **3.2 Analyze User Environment**

Responsibility: Project Manager/Team

Description: A thorough understanding of the current users' environment is necessary to define the objectives, scope, and high-level requirements of the project. Analyze the users' manual procedures or automated processes to understand what users do, how they do it, and what improvements are desired or needed. This includes gaining an understanding of the functions performed, identifying information flows within the processes, and listing process inputs and outputs.

Use appropriate data collection techniques such as user surveys, interviews, and document inspections to gather data and analyze the user environment.

Types of Information: The following list provides samples of the type of information that should be considered.

- Mission - Describe the mission of the primary user organization(s) and how the organizational mission fits into the Departmental mission and strategic plans.
- Work Processes - Analyze the work processes or tasks that are performed by the users. Identify the relationships and priority of the processes.
- Workload - Describe the volume of work currently being performed. For automated processes include processing time for batch operations, response times, peak number of simultaneous users of interactive systems, and number of transactions.
- Processing/Data Flow - Analyze the major processing/data flows for the work processes. Include the flow of data between different user groups, manual and automated processes, and different user sites.
- Integration/Interfaces - Identify interactions and interfaces that the users' current IT systems share with other IT systems.
- Users - Identify the skill sets in order to assess the capability of the organization and number of personnel at both the owner’s site and other DOE sites for Federal and contractor staff who operate, maintain, and use the current manual procedures or automated processes.

Types of Information, continued:

- Costs - Itemize costs incurred in operating the users' current manual or IT systems.

- Equipment - Identify equipment used in the current manual or IT systems and relate equipment to the function it supports in the systems.

$ Communications - Identify the guidelines, standards, equipment and software to support system communications.

- Software - Identify software packages that are being used.

$ Statutory Requirements - Identify the guidelines, directives and standards the user must comply with in the performance of the work processes.

Work Product: Develop a description of the user environment and place a copy in the Project File. The description will be incorporated into future work products such as the Project Plan and the Requirements Specifications.

Review Process: A peer review or structured walkthrough(s) may be conducted on the user environment description to ensure all pertinent information has been captured.

Activity: **3.3 Define Project Objectives**

Responsibility: Project Manager/Team

Description: Use the information gathered during the analysis of the current user environment to define the objectives of the project. The objectives should identify what the project is intended to accomplish and why it is being undertaken (e.g., to resolve problems or to satisfy statutory requirements.) Include a description of any deficiencies in the current manual and automated processes, the severity and impact of any problems, and the solutions and benefits that will result from implementing the project. The objectives should be identified in measurable terms.

Sample Questions: The following list provides sample questions that can be used to help define the project objectives. Even though the users' answers to some questions might be tentative, partial answers will be useful at this stage of the lifecycle. These questions can be revisited during the Requirements Definition Stage to help develop the project requirements.

- What is the general intent of the product?
- What organizational or Departmental functions will the product support?
- What are the major functional components of the product?
- Will the product produce any files or reports or provide data for other Government agencies, organizations, applications?
- Will the product use any data, files, or reports generated by other Government agencies, organizations, applications?
- What Departmental mission(s) will the product support?
- What Departmental strategic goal(s) will the product support?
- Will the product be aligned with the Departmental Information Management (IM) plans?
- Will the product satisfy statutory or regulatory requirements?
- What are the anticipated benefits of the product?

Sample Questions, continued:

- $ Is the product in alignment with applicable information and system architecture and architectural guidance, including the Departmental Information Architecture?

Work Product: Develop a formal statement of project need and objectives. This statement will be incorporated into the Project Plan. If a feasibility study is conducted, the statement of project objectives should be included in the Feasibility Study Document. Place a copy of the project objectives in the Project File.

Review Process: A peer review may be conducted on the formal statement of project objectives although, once the Project Plan is developed, a structured walkthrough will be conducted.

Reference: A description of the peer review process is provided in *section 2.5 Quality Reviews*.

Activity: **3.4 Define Project Scope**

Responsibility: Project Manager/Team

Description: The project scope details what user processes, organizations, and functions will be affected by the product. It also identifies the anticipated changes to current automated and manual processes. A thorough understanding of the scope of the project is necessary to determine whether the project is feasible. The scope may need to be downsized to remain feasible within the constraints of resources, budget, and time negotiated with the system owner.

Sample Questions: The following list provides sample questions that can be used to help determine the project scope. Even though the users' answers to some questions might be tentative, partial answers will be useful at this stage of the lifecycle.

- How many Government/contractor employees will use the product?
- What are the locations of the employees who will use the product?
- What tasks will be performed using the product?
- What will be the operating schedule for the product?
- How many and what types of reports will be needed?
- How and when will reports be distributed? Who receives the reports?
- What query capabilities are needed?
- Are major changes in requirements anticipated in the next few years?
- Are major changes in level of use anticipated in the next few years?
- What is the estimated life expectancy of the product?
- What are the security requirements for the product?
- What are the standards and guidance with which the product must comply (e.g., site and Departmental guidance and standards)?
- Will the product be mission-essential to DOE or mission-critical for the system owner?

Sample Questions, continued:

- Will the product contain vital records?
- What are the disaster recovery requirements for the product?
- Will the product be a stand-alone system, networked, or web-based?

Work Product: Develop a formal statement of project scope and usage. This statement will be incorporated into the Project Plan. If a feasibility study is conducted, the statement of project scope should be included in the Feasibility Study Document. Place a copy of the project scope in the Project File.

Review Process: A peer review may be conducted on the formal statement of project scope although, once the Project Plan is developed, a structured walkthrough will be conducted.

Reference: A description of the peer review process is provided in *section 2.5 Quality Reviews*.

Activity: **3.5 Develop High-Level Project Requirements**

Responsibility: Project Manager/Team

Description: High-level requirements should be of sufficient detail to make a preliminary determination about the feasibility of the project, to estimate the resources that are needed, to assess hardware and software requirements, and to estimate the need for equipment or software training.

The current and anticipated needs of all user groups must be identified. Users in different organizational units or geographic locations may have diverse or unique requirements that must be incorporated into the project requirements.

The project team participates by providing technical assistance, (i.e., to determine the validity of high-level customer/client information system requirements and that they can be accomplished.) The team also provides input by identifying concerns (implicit requirements) that will surface when customer requirements are implemented, (i.e., hardware, software, costs (indirect/direct), people resources, or training.)

System Retirement: When high-level requirements indicate an existing system no longer meets the business needs and must be replaced, the existing system will need to be retired. Reference the *Computer Systems Retirement Guidelines*, available on the Software Quality & Systems Engineering web site, for the process and forms for retiring a system.

Sample Requirements: Organize high-level project requirements into categories of related data. The following list provides samples of the types of data that should be considered.

- Inputs - Identify source documents and data that will be used as input to the processes. Provide descriptive information about data such as the type, volume, condition (e.g., edited or unedited,) organization, and frequency. Include inputs such as records or batch files from other systems that will be downloaded or migrated.

- Outputs - Identify outputs such as reports, display screens, documents, and data files.

- Databases - Estimate the high-level contents, purpose, use, format, organization, and update frequency of databases that will be used by the product. Identify other existing or planned databases that would interface with the product as a provider or recipient of information.

- Processing/Data Flow - Describe the major processing/data flow for the product. Include flow of data from the product to other systems and vice versa.

Sample Requirements, continued:

- Data Communications - Estimate the major data communications resources required to support the product. Include requirements for networks, dial-up access, and other communication configurations to support data access and retrieval requirements.

- Interfaces - Identify any systems with which the product must interface. Describe factors that may impact the design of the product.

- Security, Privacy, and Control - State requirements for ensuring the integrity of the data, for safeguarding against unauthorized access to the databases, and for other user access controls to assure that personal information about individuals collected by DOE is limited to that which is legally authorized and necessary and is maintained in a manner which precludes unwarranted intrusions upon individual privacy.

- Standards and Guidance - Describe the system or process actions or data attributes that are needed to comply with standards and guidance.

- Training - Identify the type of training required to ensure efficient operation of the product. Provide estimates of the number of personnel to be trained by type and frequency of training.

- Workload - Estimate the volume of work to be handled at slow, normal, and peak periods. Identify dates associated with each period. Include processing time for batch systems, response times, peak number of simultaneous users of interactive systems, and number of transactions.

- Costs - Estimate initial development costs and expected operating costs over the expected lifetime of the product.

- Equipment - Estimate new equipment that might need to be acquired or manufactured and current equipment that would continue to be used. Determine critical network resources.

- Software - Estimate software and firmware packages that might need to be acquired and any updates needed for existing software.

- Documentation - Determine documentation needs based on *Exhibit 2.0-1*, site-specific requirements, and other needs unique to the project.

- Usability - Determine the ease with which a user can learn to operate, prepare inputs for, and interpret outputs of a system or component.

Sample Requirements, continued:

- Operability - Define requirements associated with the operation of the system or application. The objective is to develop an IT system that requires minimal operator intervention after initial setup. Requirements should address areas such as unattended operations, automated scheduling of system or application tasks, remote intervention capabilities, operational documents, special conditions under which system must operate, (e.g., error handling and message handling for system failure and recovery.)

- Maintainability - Determine the ease with which a system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment. Determine the ease with which the system can be retained in, or restored to, a state in which it can perform its required functions.

Work Product: Develop a formal statement of the high-level project requirements. This statement will be incorporated into the Project Plan. If a feasibility study is conducted, the statement of requirements should be included in the Feasibility Study Document. The high-level requirements will serve as the foundation for the requirements developed during the Requirements Definition Stage. Place a copy of the high-level requirements in the Project File.

Review Process: A peer review may be conducted on the formal statement of high-level project requirements although, once the Project Plan is developed, a structured walkthrough will be conducted.

Reference: A description of the peer review process is provided in *section 2.5 Quality Reviews*.

Resource: The system owner organization's information resource management long-range plan provides useful planning information for consideration when developing the requirements.

Activity: **3.6 Establish Communication With Functional Areas**

Responsibility: Project Manager

Description: Early contact with the functional areas (e.g., operations, security, budget, documentation, etc.) that will provide input to, or support for, the project is necessary for developing accurate estimates of the project scope, cost, resources, and schedule. Representatives of these functional areas should be involved in all stages of the project lifecycle and are participants in the Stage Exit process.

Develop a brief profile about the project. Provide enough information so that the points-of-contact in each functional area will be able to estimate support requirements and resource allocations for the project. A sample project planning questionnaire that can be used for creating the project profile is provided at the end of this section.

Develop a list of all functional areas and points-of-contact who will provide input to, or support the project. Send each point-of-contact the project profile and request input from all recipients.

Note: This activity is not necessary for small information systems engineering projects that do not require input from other functional areas.

Work Products: Place a copy of the project profile in the Project File. Update the project profile as needed to maintain an accurate description of the product.

Keep the list of functional area contacts current and maintain a copy in the Project File and the Project Plan. Use this list as the starting point when functional areas need to be contacted about involvement in project activities such as Stage Exits.

Review Process: A peer review of the Project Planning Questionnaire is optional; however, it is encouraged as it may provide helpful insight.

Reference: Information on the identification of functional areas is provided in the DOE *Stage Exit Process Guide*, available on the Software Quality & Systems Engineering web site.

Resources: Information on the identification of functional areas can be found in the Project Plan example template on the Software Quality & Systems Engineering web site.

A downloadable copy of the Project Planning Questionnaire is available on the Software Quality & Systems Engineering web site.

Exhibit 3.6-1. Project Planning Questionnaire

Objectives of this questionnaire: Enable project teams (immediate and extended) to be cognizant of the disparate planning activities, which can affect project outcome. Provide early notification to the stakeholders that a new project may involve their area. Document information to help develop resource estimates and identify risks.

Project Planning Questionnaire

Project managers: *Please complete and distribute this questionnaire to all project stakeholders (e.g., system owners, users, computer operations, telecommunications, LAN engineering, training, security, standards, documentation); see last page of questionnaire for additional guidance on developing a distribution list. Provide as many answers as are possible at this time; the questionnaire is not designed to capture detailed requirements. Consider using this questionnaire as a basis for a project startup meeting.*

If the project is not started immediately after the questionnaire is distributed, redistribute at project startup to provide current information. All areas should be notified of the project well in advance of scheduling the first formal checkpoint, (e.g., the Planning Stage Exit, as defined in the DOE Systems Engineering Methodology.)

Recipients of the questionnaire: *Please review as soon as possible and provide feedback as appropriate to the project manager.*

Project Profile

Project Name & Acronym _____

Contract No. _____ Project No. _____

Is there separate funding for this project? Yes _____ No _____

DOE User Organization _____ DOE Federal POC _____
(e.g., Program Manager, Technical Monitor)

E-Mail address _____ Phone _____

Project Manager & organization _____
(Federal or contractor. The person who has daily responsibility for planning, tracking, and controlling the project))

E-Mail address _____ Phone _____

Project Profile (continued)

A. Are measurable business objectives documented as goals for this project?

Name & Date of document _____

Provide a brief description of what this project will accomplish. (i.e. the project objectives)

_____

_____

B. What size effort (e.g., as defined by the SEM[2]) do you anticipate this project to be?

Small _____ Medium _____ Large _____

C. Where are the users located? (check all that apply and/or fill-in information)

___ DOE HQ ___ Operations Office ___ Field Office
___ Area Office ___ Laboratory ___ Power Administration

Name of DOE site, laboratory, university, or collaborative partnership (e.g., between a laboratory or university of the US and that of another country)

_____

Other _____

D. List any known assumptions, constraints, and dependencies associated with this project.

_____

_____

E. List any known risks associated with this project.

_____

_____

[2] Reference, DOE SEM, Exhibit 2.1-1, Information Systems Project Sizes

Project Information

For each question, check all that apply, and supply specific information where requested.

1. This project involves: (check all that apply)

___ System Development
___ Software Development
___ Commercial Off The Shelf (COTS)
___ Customization (of COTS)
___ System Maintenance
___ Software Maintenance
___ Infrastructure Upgrade(s)
___ Major System Enhancement

Other

2. Platform - Software: (check all that apply)

___ Client-Server ___ Distributed ___ Stand Alone ___ Web Enabled

Other _____

3. Platform - Hardware: (check all that apply)

___ Mainframe ___ Desktop Computer ___ Dedicated Server ___ Shared Server

Other e.g., Enterprise Server_____

4. Infrastructure; the target IT environment: (check all that apply)

___ Backbone ___ Organizational LAN ___ Mainframe ___ Organizational WAN
___ Corporate[3] WAN ___ Other _____

Name of WAN/LAN/Server (if known) _____

5. Network Topology.

___ Ethernet ___ Token Ring ___ ATM

Other _____

[3] Reference DOE Information Architecture, Volume 4, Vision, page 3-20

Project Information (continued)

6. Communications Protocol.

___ TCP/IP (Unix,NT) ___ IPX/SPX (Novell) ___ NetBios (IBM)
___ Vines (Banyan Vines)

Other _____

7. Network environment required to support the application.

___ Peer ___ Single server ___ Multi server ___ Enterprise

Other _____

8. The following are examples of current information systems technology, and not an exhaustive list. Please fill in your particular technology if it does not appear. (check all that apply)

___ Data Mining
___ Workgroup Computing
___ E-Mail / Voice-Mail
___ Distributed Application Architecture
___ Push Desktop
___ Wireless Communications
___ 3 Tier Model Architecture
___ _____
___ _____
___ _____

___ Data Warehouse
___ Network Management
___ Remote Access
___ Distributed Video Conferencing
___ Smart Card
___ Modular Software Components
___ Internet Access and Support
___ _____
___ _____
___ _____

9. **Infrastructure Impact Assessment - Local Backbone** (fill in requested information on the first line and check all that apply)

Hours of operation _____ Number of simultaneous users _____
___ Connectivity exists to all identified users
___ Encryption required for all Messages/Packets/Transactions

The following are associated with Transporting Description:
___ Text only ___ Graphics only ___ Text and graphics
___ Multimedia ___ Voice ___ Video
___ Transaction oriented ___ File oriented

Project Information (continued)

10. Infrastructure Impact Assessment - Wide-Area Backbone (fill in requested information on the first line and check all that apply)

Hours of operation _____ Number of simultaneous users _____
___ Connectivity exists to all identified users
___ Encryption required for all Messages/Packets/Transactions

The following are associated with Transporting Description:
___ Text only ___ Graphics only ___ Text and graphics
___ Multimedia ___ Voice ___ Video
___ Transaction oriented ___ File oriented

11. Operating Environment(s): (check all that apply and fill the version and release number)
V=Version R=Release

___ MS Windows, V.R _____ ___ OS/2, V.R _____ ___ Unix/AIX, V.R _____
___ Windows N/T, V.R _____ ___ VM, V.R _____ ___ Macintosh, V.R _____
___ Novell, V.R _____ ___ MVS, V.R _____ ___ CICS, V.R _____
___ DOS, V.R _____ ___ Sun, V.R _____ ____ Linux, V.R _____
___ Web Browser (specify) _____, V.R _____

Other _____, V.R _____

11a. The following constraints exist in the targeted operating environment(s) (check all that apply)

___ Firewalls ___ Transmission speed ___ Server capacity
___ Security ___ Workstation capacity ___ Remote access capability
___ Widely dispersed user community without appropriate inter-operability

Other _____

12. Identify the development languages that will be, or are being considered for use (check all that apply and fill-in the version and release number) V=Version R=Release

___ Cobol, V.R _____ ___ Java, V.R _____ ___ C++, V.R _____
___ Visual Basic, V.R _____ ___ FoxPro, V.R _____ ___ Paradox, V.R _____
___ HTML, V.R _____ ___ Delphi, V.R _____ ___ Visual C++, V.R _____
___ Power Builder, V.R _____ Other _____, V.R _____

Project Information (continued)

13. What is the sensitivity of the project's data?

___ Classified ___ Unclassified Sensitive ___ Unclassified Non-Sensitive

13a. Which protective means are required[4] ? (check all that apply)

___ Firewall ___ Encryption ___ Token ___ Digital Signature
___ Detection ___ Authentication ___ Validation
___ User controlled ID and password ___ Removable hard drive
___ User ID & password, controlled by system administrator ___ Physical Security
___ Read/write access, controlled by system administrator ___ Biometrics

Other _____

14. List the database management systems that will be, or are being considered for use.

___ Oracle ___ Sybase ___ MS Access ___ DB2

Other _____

15. If this is a Corporate system, who is going to own the function for the corporate data administration?
(e.g., J. White, Controller; S. Black, Chief Financial Officer; T. Auburn, Director of HR)

Name & Function/Position _____

16. Is the system or application's operation so essential that data must be immediately available at all times or recoverable within a short time frame?

___ No ___ Yes

If you answered yes, have you considered requirements for[5]: (check all that apply)

___ Data recovery ___ Backups ___ Fault Tolerance
___ System Performance ___ Mirroring/Imaging ___ Disaster Recovery (crisis management)

[4] Reference DOE HQ Computer Protection Plan, Chapter 4, page 4-1

[5] Reference DOE Directives, 151, 200, 1360, and 471 (www.directives.doe.gov)

Project Information (continued)

17. List existing systems, applications, or data sources, if any, with which this system will interface (sharing data, receiving data from, or sending data to).

System/Application	System/Data Source
_____	_____
_____	_____

Project Management Factors

Answers to the following questions will help provide a better understanding of how information systems project management practices are implemented within the Department.

18. Which lifecycle methodology will be followed on this project?

___ DOE Systems Engineering Methodology (SEM)
___ Microsoft Solutions Framework
___ James Martin

Other e.g., IBM Proprietary

19. The Software Engineering Institute's (SEI) Software Capability Maturity Model[6] (CMM) identifies the relative maturity of software development/maintenance organizations. The CMM level of our organization is:

___ 1 ___ 2 ___ 3 ___ 4 ___ 5
___ No evaluation has been performed.

20. Which of the following software development/maintenance techniques (if any) do you plan to use? (check all that apply)

___ Rapid Application Development (RAD)
___ Joint Application Development (JAD)
___ Segmented Development
___ Spiral
___ Rapid Prototyping
___ Iterative Development
___ Object oriented
____ Waterfall

Other _____

[6] For additional information, visit the SEI Web site at http://www.sei.cmu.edu

Project Management Factors (continued)

21. Which individual, team, and organizational information systems development and management practices will be implemented on this project? (check all that apply)

___ Project Planning
___ Configuration Management
___ Quality Assurance
___ Risk Assessment
___ Training Program
___ Inter-group Coordination
___ Organizational Process Definition
___ Defect Tracking
___ Requirements Management
___ Project Tracking and Oversight
___ Sub-Contractor Management
___ Peer Reviews
___ Software Product Engineering
___ Integrated Software Management
___ Organizational Process Focus

Other _____

22. What training do you anticipate using to complete this project? (check all that apply)

___ Individual ___ Classroom ___ Self Study ___ CD Rom
___ Video Tape ___ I-TV ___ Computer Based Training (CBT)

Other _____

23. Which medium will be used for the project/system documentation? (check all that apply)

___ Web Site ___ Hardcopy ___ CD Rom ___ Online Help
___ Video ___ Quick Reference Card ___ Diskette

Other _____

24. What measurements will be used to track this project? (check all that apply)

___ Cost ___ Schedule ___ Effort ___ LOC
___ Defects ___ Function Points

Other _____

Questionnaire Completed by

Name _____

Organization _____ **Date** _____

Distribution List

In general, the completed questionnaire should be sent to all immediate and extended project team members, stakeholders, and others with a need to know. Refer to the *Stage Exit Guide*, Examples 2, 3, 4, and 5, for guidance on preparing a distribution list.

Optional

A peer review, (e.g., conducted by someone who has been involved with a similar project,) can provide helpful insight and help improve the quality of the questionnaire and the project. Has an independent peer review been conducted on this completed questionnaire?

___ Yes ___ No

If Yes, conducted by:

Name _____ **Date** _____

Name _____ **Date** _____

Activity: **3.7 Determine Project Feasibility**

Responsibility: Project Manager/Team

Description: The feasibility of successfully developing and implementing the project is determined in the Planning Stage. Software and hardware alternatives are reviewed and used to formulate preliminary platform options. Project feasibility leads to a "go" or "no go" decision about the project. Determining project feasibility is an interactive process of collecting and analyzing data and searching for cost-effective, viable technical solutions.

Use the project objectives, scope, and high-level requirements as the basis for determining project feasibility. Work with the user organization and functional area representatives to address technical issues and risks. Conduct research and investigate documents and other resources. When determining project feasibility, a brief feasibility review may be all that is required; however, when a proposed solution is mandated to define risks and challenges to successfully complete a project, an in-depth feasibility study with a detailed analysis of benefits and costs may be required.

Note: Feasibility may not be an issue for some small development projects. A Feasibility Review is not required when feasibility is obvious.

Sample Questions: The following is a list of sample questions that can be used to help determine the feasibility of a project.

- Can the users needs/problems best be satisfied with a manual process, automated process, or combination?
- Is it cost effective to develop an automated process?
- Is the scope of the project feasible within time, resource, and hardware and software constraints and limitations?
- Is there at least one technically feasible IT solution for the project?
    - If a project is well defined and has no automation issues, a single straightforward IT solution may sufficiently demonstrate cost and technical feasibility.

Sample Questions, continued:

- Where IT issues have been identified, technical alternatives should be associated with each proposed solution.

Work Products: Refer to each task for applicable work products.

Review Process: Refer to each task for applicable review processes.

Tasks: The following tasks are involved in determining project feasibility.

3.7.1 Investigate Software Alternatives
3.7.2 Investigate Hardware Alternatives
3.7.3 Formulate Platform Options
3.7.4 Conduct Feasibility Review
3.7.5 Conduct Analysis of Benefits and Costs
3.7.6 Conduct Feasibility Study

Task: **3.7.1 Investigate Software Alternatives**

Description: When the software to be used for the project has not been predetermined by the system owner's existing IT environment, software available within the Department and the commercial marketplace should be investigated. In the Planning Stage, the investigation of software alternatives is geared to determining project feasibility.

Unless the cost effectiveness of developing custom-built software to meet mission needs is clear and documented, all sources of reusable code, applications, and commercial-off-the-shelf (COTS) software must be investigated on a site and Departmentwide basis prior to making a decision to custom build code for the project. This practice ensures the most cost-effective and efficient use of resources, and will decrease the number of duplicative and overlapping software systems. The choice to develop a customized application should be balanced against the availability of other solutions; and the project cost, resources, and time constraints.

Software Alternatives: Information on software products or modules can be obtained by notifying field sites, DOE Headquarters, other Government agencies, and private industry via the Internet. The following is a list of software alternatives that should be considered.

- Adapt existing software in use within the Department.
- Adapt existing software in use within other Government agencies.
- Adapt mainframe or minicomputer source code obtained from existing DOE systems.
- Purchase commercial-off-the-shelf (COTS) software.
- Reuse existing modules of code.
- Adapt reusable code to fit the new application.
- Develop a custom-built software product.

Note: Medium and small information systems engineering efforts are often restricted to the system owner's existing software. This should not preclude the potential cost savings of re-engineering existing software modules rather than custom building the entire software system.

Task: **3.7.2 Investigate Hardware Alternatives**

Description: When the hardware to be used for the project has not been predetermined by the system owner's existing IT environment, investigate hardware available within the Department and through the commercial marketplace. In the Planning Stage, the investigation of hardware is geared to determining project feasibility.

Factors to Consider: The following is a list of factors that should be considered when identifying hardware alternatives.

$ Availability and cost of hardware
- Shareable hardware
- Government excess
- New procurement

$ Architecture compliance

$ Current and future communications needs

$ Computer security requirements of the system

$ Volume of data

$ Importance of data to the Departmental mission

$ Importance of data to the user organization's mission and to job performance

$ Potential growth to serve more users

$ Potential growth to serve more locations

$ Interface to other systems or organizations

$ Conformance to Government standards such as networking and open systems

Note: Medium and small systems engineering efforts are often restricted to the system owner's or user sites' existing hardware.

Task: **3.7.3 Formulate Platform Options**

Description: Use the information collected about software and hardware alternatives to formulate preliminary platform options. The purpose of identifying platform options early in the project lifecycle is to assure that at least one technically feasible and cost-effective approach exists to satisfy the project objectives. If more than one platform option is feasible, identify the benefits, costs, assumptions, constraints, dependencies, and risks associated with each option. Platform options must be compatible with the DOE target Enterprise Architecture (EA).

No platform decisions are made at this time. Detailed technical solutions are premature prior to defining the product requirements. The platform alternatives information gathered in the Planning Stage is revisited in the Functional Design Stage, at which time a final recommendation is developed by the project team and presented to the system owner. The system owner is responsible for making the final platform decision.

Work Product: Develop a summary of platform options for use in the Feasibility Review or Feasibility Study. Place a copy of the platform option information in the Project File.

Review Process: Conduct a structured walkthrough to ensure that the most viable platform options have been identified.

Task: **3.7.4 Conduct Feasibility Review**

Description: A Feasibility Review is a meeting to determine whether the project can be accomplished with the available resources, system owner and user's IT environment, and technological constraints. The Feasibility Review meeting also provides an opportunity for project management to obtain feedback from other project managers and the functional area representatives who will be providing input to, or supporting, the project throughout the lifecycle.

The project objectives, scope, high-level requirements, and preliminary platform options should be shared with the review meeting participants prior to the meeting date. The participants are expected to evaluate the project information and risks, and make a recommendation (i.e., feasibility statement) about project feasibility.

Feasibility Factors: The following are some typical factors that should be considered when determining the feasibility of a project.

$ Project scope and objectives
$ Users' IT environment
$ High-level requirements
$ Assumptions, constraints, and limitations
$ Platform options
$ Security and recovery objectives
$ Risk factors
$ Technological factors
$ Available resources and budget
$ Future growth needs
$ Expected long-term benefits
$ Compliance with long-range information resource management plans

Recommendations: After all of the pertinent feasibility factors have been considered, the review meeting participants should make one of the following recommendations:

- Proceed with the project without performing a Feasibility Study
- Prepare an Analysis of Benefits and Costs (business case)
- Conduct a Feasibility Study, which includes an Analysis of Benefits and Costs (business case)
- Stop the project

Work Products: Generate a record of the Feasibility Review meeting to serve as verification that the review occurred, to record feasibility factors that were considered and the recommendation(s) generated during the meeting, and to provide background information if a Feasibility Study or Analysis of Benefits and Costs is required.

The project manager uses the recommendations from the Feasibility Review meeting to develop a formal statement of feasibility. A typical statement of feasibility is a short declaration describing whether or not it is feasible to develop the project within the known constraints. When major risks are involved in the feasibility decision, it may be necessary to develop a separate risk analysis report that describes the risk factors and their consequences.

Depending on the factors that must be considered for each project, the statement of feasibility may contain the following information.

- Project objectives
- Summary of issues concerning:
  - development and implementation
  - assumptions, constraints, and limitations
  - project scope
- Results of research on hardware and software alternatives
- Significant risk factors
- Feasibility recommendation(s)

The project manager decides on the final recommendation and reports the findings to the system owner for review and approval. Place a copy of the findings in the Project File.

Sample Feasibility Statement: The following is a sample feasibility statement for a low-risk project that would use the hardware/software platform currently available within the users' organization.

The client organization, Program point of contact, Management Officer, and project manager agree that the XYZ project will be written in {development language} and use {operating system and/or DBMS} on {hardware configuration}, all of which are currently in place and can easily absorb the impact of XYZ. This will be a custom-built product since a search of software repositories did not reveal any reusable or existing software that would satisfy the project requirements.

Task: **3.7.5 Conduct Analysis of Benefits and Costs (as appropriate)**

Description: An Analysis of Benefits and Costs (ABC) is a useful tool in any stage of the lifecycle. In the Planning Stage, the results of an ABC help to determine the feasibility of a project and the return on investment. For example, an ABC can be conducted to determine if changing the users' current business processes or IT environment will improve efficiency or reduce overhead expenditures enough to justify the cost of the project, and when the system owner can expect to recoup the costs of the project in benefits.

An ABC is used to identify and compare the benefits and costs associated with all of the hardware or software alternatives. Any advantage to a particular alternative is considered a benefit, and any loss or penalty is considered a cost. Costs can also include the purchase price of supplies, equipment, software, personnel time or charge rate, and system downtime. The results of the ABC indicate the most cost-effective alternative.

When a totally manual process is being automated, the benefits of automating the process may be obvious. If the system owner has restricted the platform, then an ABC can be an appropriate way to document these decisions and the benefits and costs associated with the limitations.

When a Feasibility Study is performed, an ABC is a mandatory requirement of the study. When a Feasibility Study is not performed, the ABC is an optional process.

Work Product: Develop a report that describes the results of the ABC. When a Feasibility Study is performed, the results of the ABC will be incorporated into the Feasibility Study Document.

A formal business case may be appropriate and/or required in addition to the ABC.

Review Process: An informal peer review or a structured walkthrough is recommended to validate the ABC process used and the results obtained.

Reference: The following Department documents provide guidance on conducting an Analysis of Benefits and Costs and can be found on the Software Quality & Systems Engineering web site.

- Analysis of Benefits and Costs (ABC's) Guideline: Volume 1, A Manager's Guide to Analysis of Benefits and Costs
- Analysis of Benefits and Costs (ABC's) Guideline: Volume 2, An Analyst's Handbook for Analysis of Benefits and Costs

Task: **3.7.6 Conduct Feasibility Study (as appropriate)**

Description: When a project faces decisions or issues that require a more detailed investigation than is possible with a Feasibility Review or Analysis of Benefits and Costs (ABC), a Feasibility Study must be performed to obtain the necessary information for making an informed decision about project feasibility.

The following are examples of cases where a Feasibility Study must be performed.

- There is uncertainty or disagreement on the boundaries of the project.
- There is uncertainty over the cost justification or technical feasibility of a project.
- There is a lack of agreement about the goals or approach for building the product.
- The proposed size or complexity of the product indicates a high degree of risk.
- The product will automate functions that currently are not being performed.

Note: New products can be limited to the system owner's and user's existing hardware and software environment, and may not require a Feasibility Study. An existing Feasibility Study can be used if the information is current, relevant to the new project, and technically correct. In cases where the platform is limited or restricted, the Feasibility Study may be abbreviated to evaluate only the technical solutions for the areas that have some flexibility.

Use the information identified in the Feasibility Review and the Analysis of Benefits and Costs as the basis for the Feasibility Study. Consider any preliminary solutions that were formulated and identify the alternative ways to resolve the problems or issues. Evaluate all of the available feasibility factors to determine if the project is technically feasible and cost effective.

Business Case: Prior to making major expenditures on corporate IT initiatives, the Office of Management and Budget (OMB) requires that Federal agencies go through an analysis process to develop a business case to justify their investment. The requirement is based on OMB's interpretation of how Federal agencies must comply in carrying out the mandates of Congressional legislation. The analysis process that DOE employs in developing a business case is called the Strategic Information Management (SIM) process, which is based on guidance from the General Accounting Office (GAO). Information on the SIM process is available on the CIO's web site.

Description, continued:

A risk analysis is a useful tool that will help to determine the feasibility of a project and the potential to succeed. Conduct a risk analysis (identify, assess, and document) based on the cost, resources, schedule, and technical aspects of the feasibility study.

- Risks should be analyzed and prioritized based on their potential impact to the project
- Contingencies for the risks are identified

Work Product:

The results of the Feasibility Study are reported in a document that describes the process that was used to determine feasibility, the alternatives that were considered, and the results of the Analysis of Benefits and Costs. The Feasibility Study results determine the feasibility recommendation for the project. More detail on work products can be found within the applicable subtask.

Review Process:

An informal peer review or a structured walkthrough is recommended to validate the Feasibility Study process used and the results obtained. More detail on review processes can be found within the applicable subtask.

Reference & Resources:

The following references and resources are available on the Software Quality & Systems Engineering web site:

- The Feasibility Study template
- *Analysis of Benefits and Costs (ABC's) Guideline: Volume 1, A Manager's Guide to Analysis of Benefits and Costs*
- *Analysis of Benefits and Costs (ABC's) Guideline: Volume 2, An Analyst's Handbook for Analysis of Benefits and Costs*

Subtasks:

The following subtasks are involved in conducting and documenting a Feasibility Study.

3.7.6.1 Analyze the Alternatives
3.7.6.2 Determine Feasibility Recommendation
3.7.6.3 Develop Feasibility Study Document

Subtask: **3.7.6.1 Analyze the Alternatives**

Description: Alternatives for developing and implementing a project are derived from the high-level project requirements, the results of the Feasibility Review and the Analysis of Benefits and Costs, and the preliminary platform options. The analysis of the alternatives forms the basis for determining project feasibility.

The analysis of the alternatives should consider the following types of information.

- The ability of each alternative to achieve the project objectives.
- The ability of each alternative to meet the users' requirements and expectations.
- How well each alternative accommodates the system owner's current processes and resources.
- How cost-effective and technically feasible each alternative is compared to the existing automated or manual process.
- How well each alternative fits with the hardware and software limitations imposed by the system owner.

Analysis of the alternatives may include the following activities.

- Research current computer industry periodicals and web sites to obtain articles and reviews about software and hardware alternatives.
- Interview software and hardware vendors to obtain up-to-date information about product releases and future upgrades, capabilities, vendor support, developer training, product demonstrations, multi-user license arrangements, current users, and costs.
- Interview current users of the product to obtain information about user satisfaction, ease-of-use, satisfaction of user expectations, productivity, and product limitations.

Subtask: **3.7.6.2 Determine Feasibility Recommendations**

Description: The results of the Feasibility Study are used to determine project feasibility. The feasibility recommendations must be substantiated by the results of the Analysis of Benefits and Costs (ABC).

The feasibility recommendations should include the following types of information.

$ The recommended alternative for each of the project automation issues.

- The feasibility to develop the project.
- The most technically sound alternative with the most long-range benefits to the Department.
- The most cost-effective configuration for the project based on the ABC.
- The estimated total lifecycle costs based on the recommended technical solution and the ABC.

Subtask: **3.7.6.3 Develop Feasibility Study Document**

Description: The Feasibility Study Document provides the following types of information.

- The process that was used to determine project feasibility.
- The alternative approaches that were analyzed for achieving the project objectives.
- The results of the Analysis of Benefits and Costs.
- The recommendations for a specific approach to meet the system owners' and users' project objectives, automation needs, and expectations.

Work Product: The Feasibility Study Document should contain enough information to enable the system owner to make a decision to either continue or terminate the project.

Review Process: An informal peer review or a structured walkthrough is recommended to validate the Feasibility Study Document and feasibility recommendations.

The completion of the Feasibility Study is an appropriate time to schedule an In-Stage Assessment (ISA).

Reference: *In-Stage Assessment Process Guide* provides a procedure and sample report form that can be used for in-stage assessments.

Resources: The following resources are available on the Software Quality & Systems Engineering web site:

$ The Feasibility Study template

Activity: **3.8 Develop Project Plan**

Responsibility: Project Manager

Description: The purpose of the Project Plan (sometimes called a Software Development Plan) is to establish reasonable plans for performing the information systems engineering activities and for managing and tracking the project. The following project management activities and lifecycle activities described in this methodology provide input for the Project Plan. The project management activities are initiated in the early stages of, and in parallel with, the overall project planning.

$ Define the management approach for the project including project tracking and oversight activities.

$ Formulate the technical approach for the project.

$ Determine the project standards to be used in developing the plan, (e.g., customer standards, project standards, or approved statements of work,) and determine the selected procedures, methods and project standards for developing and maintaining the system.

$ Collect the information needed to develop the project estimates and assess their reasonableness.

$ Prepare an estimate of capacity requirements for the systems engineering facilities and support tools that are based on the size estimates of the work products and other characteristics. Examples of facilities and support tools include:

- Development computers and peripherals
- Test computers and peripherals
- Target computer environment
- Other support software

Responsibilities are assigned and commitments are negotiated to procure or develop these facilities and support tools. The support efforts are budgeted and agreements (charters) are documented. Plans are reviewed by all affected groups.

Description, continued:

- Establish the project development team, (i.e., systems analysts, developers, QA, CM, and documentation support,) and coordinate and negotiate Plans with the project team.

- Identify the training needs of each member of the project team. This should include skills required, by whom, and when they are required. Develop strategies and plans for providing the needed training to those individuals that require it. This effort should be aligned with the Training Plan defined in *Section 1.2.2,Organizational Training*.

- Establish the project schedule, (i.e., a lifecycle with predefined stages of manageable size).

- Managed and control the Project Plan. The plan is reviewed by:

  - Project Manager
  - Project Software Manager
  - Other system Managers
  - Other affected groups

Note: A Project Plan is an effective management tool that is recommended for all projects regardless of size. The lifecycle stages documented in the plan can be consolidated for small projects.

Reference: Chapter 2.0, Lifecycle Model, *Adapting the Lifecycle*, discusses lifecycle issues in relation to developing a Project Plan.

Resources: The following resources are available from the Software Quality & Systems Engineering web site to assist with the development of a Project Plan.

- A Project Plan Example Document that provides a plan for a fictitious project.

- Three project models were created in Microsoft Project to support the development of the project schedule. The project models provide detailed work breakdown structures for (1) a basic model, (2) a model that follows the Project Plan Example, and (3) an expanded model that includes recurring management activities such as structured walkthroughs.

The Institute of Electrical and Electronics Engineers Standard for Software Project Management Plans (Std 1058.1-1998) also provides guidance on developing project plans.

Work Product: Develop a Project Plan that provides detail for the Planning and Requirements Definition Stages and high-level information for the other lifecycle stages. At the conclusion of each stage, review the Project Plan to determine if the project estimates for resources, cost, and schedule need to be revised for either the current stage or subsequent stages. The updated Project Plan is reviewed as part of each stage's quality reviews (i.e., Structured Walkthroughs, In-Stage Assessment, and Stage Exit). In addition, the Project Plan will be expanded to provide detailed estimates of resources, costs, and hours for the next stage.

Some of the critical elements of a Project Plan include:

- Project Overview including list of deliverables
- Project Organization including organizational structure, project responsibilities, and process model
- Management Process including assumptions, dependencies, and constraints; risk management; tracking and oversight, and staff planning
- Training to be provided to project personnel, plans and strategies for providing the training, estimate of training costs, and training schedule
- Technical Approach including development methodology, other methods, tools, techniques, project documentation, and support functions
- Work Packages, Schedule and Budget including resource requirements, budget allocations, schedule, and work breakdown structure

Review Process: Conduct a structured walkthrough to ensure that the Project Plan reflects the project objectives and scope, identifies and mitigates project risks, and adequately estimates the project resources, costs, and schedule. The Project Plan should be reviewed by the project manager and other affected groups.

Tasks: The following tasks are involved in developing the project plan.

3.8.1 Develop the Project Estimate
3.8.2 Develop the Project Schedule
3.8.3 Perform Project Tracking and Oversight

Task: **3.8.1 Develop the Project Estimate**

Description: When developing the Project Plan, duration of activities, (e.g, planning, documenting, coding, or testing, and project costs,) (e.g., labor, equipment, hardware, or software,) are estimated and included in the Plan. The project estimates are used for obtaining budget and resource commitments.

More than one estimation methodology should be used for comparison and verification purposes. No one methodology is necessarily better than another, their strengths and weaknesses are often complementary to each other. One method may overlook system level activities such as integration, while another method may include integration, but overlook some key development points.

Organizational Database
Establish and maintain an organizational process database containing metrics data from completed projects. Data should include costs and duration, and should be categorized to help new projects in developing resource estimates. Categories may include areas such as functionality, user interface, database access, development languages, and defects found/removed per unit of measure.

Estimates for new projects should be developed using a combination of one or more estimating methodologies, and input from the organizational process database. Data from new projects should be captured in order to incrementally build a more robust, accurate, and diverse base of data. Project management tools selected should allow for, (i.e., "exporting" data to the organizational database.) The following paragraphs describe estimating methodologies that may be used to estimate project costs.

Analogy Method
Estimating by analogy means comparing the proposed project to previously completed similar projects where project development information is known. Actual data from the completed projects are extrapolated to estimate the proposed project. Estimating by analogy can be done either at the system level or the component level.

The main strength of this method is that the estimates are based on actual project data and past experience. Differences between completed projects and the proposed project can be identified and impacts estimated. One problem with this method is identifying those differences. This method is also limited because similar projects may not exist, or the accuracy of available historical data is suspect. Also, many projects may not have historical precedents.

Description, continued:

Bottom-Up-Method
Bottom-up estimates involve identifying and estimating individual systems development components (tasks) separately, then combining the results to produce an estimate of the entire project. It is often difficult to perform a bottom-up estimate early in the lifecycle process because the necessary information may not be available. This method also tends to be more time consuming and may not be feasible when either time or personnel are limited.

Top-Down Method
The top-down method of estimation is based on overall characteristics of the project. The project is partitioned into lower-level components and lifecycle phases beginning at the highest level. This method is more applicable to early cost estimates when only global properties are known.

Advantages include consideration of system-level activities (e.g., integration, documentation, project control, quality assurance, and configuration management), many of which may be ignored in other estimating methods. The top-down method is usually faster, easier to implement, and requires minimal project detail. Disadvantages are that it can be less accurate and tends to overlook lower-level systems development tasks and possible technical problems. It also provides very little detail for justifying decisions or estimates.

Expert Judgment Method
Expert judgment involves consulting with subject matter experts to use their experience and understanding of a proposed project to provide an estimate for the cost of the project.

The obvious advantage of this method is the expert can factor in differences between past project experiences and requirements of the proposed project. The expert can also factor in project impacts caused by new technologies, applications, and languages. Expert judgment always complements other estimation methodologies. One disadvantage is that the estimates can be no better than the expertise and judgment of the expert. It is also hard to document the factors used by the experts who contributed to the estimate.

Parametric Method
The parametric method (also referred to as the algorithmic method) involves the use of equations to perform system estimates. The equations are based on research and historical data and use such inputs as source lines of code (SLOC), number of functions to perform, and other cost drivers such as language, design methodology, skill levels, and risk assessments.

Advantages of this method include being able to generate repeatable results, easily modifying input data, easily refining and customizing formulas, and better understanding of the overall estimating methods since the formulas can be analyzed. However, the results are questionable when estimating future projects

Description, continued:

that use new technologies. The equations are generally unable to deal with exceptional conditions such as exceptional personnel in any cost estimating exercises, exceptional teamwork, and an exceptional match between skill levels and tasks. However, any estimating approach can be impacted by these scenarios, and care should be exercised when accounting for such concerns.

Task: **3.8.2 Develop the Project Schedule**

Description: The project schedule should be based on the following types of information:

- Size estimate of the work products (or the size of changes)
- Information systems effort and costs
- Imposed milestone dates, critical dependency dates, and other constraints
- Documentation assumptions made in deriving the schedule
- Coordinated documentation schedule for review and agreement

Task: **3.8.3 Perform Project Tracking and Oversight**

Description: Tracking and oversight involves the tracking and reviewing of actual project accomplishments and results against documented estimates, commitments and plans, and adjusting the original plans based on the actual accomplishments and results of the project.

Project tracking and oversight provides visibility into the actual progress so that management can make corrective actions when project performance deviates significantly from the original plans.

The planning process, which is described in the Project Plan, is used as the basis for tracking the project activities, communicating status, and revising plans. Progress is primarily determined by comparing the actual size, effort, cost and schedule to the plan when selected work products are completed and at selected milestones. Where it is determined that the project's plans are not being met, corrective actions are taken. These actions may include revising the Project Plan to reflect the actual accomplishments, and replanning the remaining work to be done and/or taking actions to improve the performance.

The results and accomplishments are reviewed with senior management where approved corrective changes are made to the original plan and communicated to the project team. Tracking and oversight addresses the following responsibilities:

- The original Project Plan is used for evaluating project activities and communicating status.
- The original Project Plan is revised, as appropriate, to incorporate significant requirements and plan changes; for example, changes effecting interdependencies between the system requirements, design constraints, resources, costs and schedule.
- Project commitments and changes made to those commitments are reviewed with senior management.
- Approved changes to commitments that affect the project are communicated to the project team.
- The size of the work products and/or size of the changes to the products are tracked and necessary corrective actions are taken.

Description, continued:

- The project's effort and costs are tracked and necessary corrective actions are taken.

$ Critical computer resources are tracked and necessary corrective actions are taken.

$ The project's schedule is tracked and necessary corrective actions are taken.

$ Systems engineering technical activities are tracked and necessary corrective actions are taken.

$ Risks associated with cost, resource, schedule, and technical aspects are tracked and senior management is advised accordingly.

$ Project management data and replanning data are recorded, (e.g., revised estimates, system planning data, and actual measurement data for future use.)

$ Measurements are made to determine status of the tracking and oversight activities.

$ Periodic internal reviews are conducted to track technical progress, planning, and performance, against the original Project Plan and the plan is revised accordingly.

$ Final reviews are conducted at selected milestones to address the accomplishments and results of the project.

Activity: **3.9 Develop Quality Assurance Plan**

Responsibility: Project Manager and Quality Assurance Manager

Description: The quality assurance program involves the reviewing and auditing of the products and activities to verify that they comply with the applicable procedures and standards and to assure the production and operation of high quality products according to stated requirements. The results of these reviews and audits provide the project manager and other support area managers with appropriate visibility into the processes used.

The quality assurance program is initiated at the beginning of a project and is conducted throughout the information systems engineering lifecycle. The quality assurance program is the joint responsibility of the project manager and quality assurance manager with direct support and involvement from the quality assurance representatives assigned to the project. Quality assurance includes the following activities:

- Preparation of a Quality Assurance Plan for the project according to a documented procedure and coordinated with the designated project team members.
- Establishment of a reporting channel regarding quality assurance.
- Allocation of adequate resources and funding to maintain and perform quality assurance activities.
- Provision of formal training for the quality assurance representatives to perform their activities.
- Provision of orientation training in the role, responsibilities, authority and value of quality assurance to project developers.
- Performance of quality assurance activities in accordance with the Quality Assurance Plan.
- Participation by quality assurance representatives in the preparation and review of the Projects Plan, standards, and procedures.
- Reviews by the quality assurance representatives of the systems engineering activities to verify compliance.

Description, continued:

- Audits by quality assurance representatives of designated work products to verify compliance.

$ Periodic reports by quality assurance representatives of the results and work products of quality assurance activities to the project team.

$ Documentation of deviations identified in the project activities and work products and handled in accordance with a documented procedure.

$ Use of measurement and analysis techniques to determine the cost and schedule status of the quality assurance activities.

$ Periodic reviews by independent quality assurance representatives of the activities and work products of the project's quality assurance program.

Quality assurance personnel work with the project manager during early stages to establish plans, standards, and procedures that will add value to the product and satisfy the constraints of the project and the organization's policies. By participating in establishing quality work products, (e.g., the plans, standards, and procedures,) the quality assurance personnel help ensure the work products fit the project's needs and verify that the work products will be usable for performing reviews and audits throughout the project lifecycle.

Compliance issues are first addressed with the project manager and resolved at that level, if possible. Issues not resolvable by the quality assurance representative and project manager are escalated to an appropriate level of management for resolution.

Work Product: The quality assurance manager or designated representative assists the project manager with the development of a plan that clearly defines the project's quality assurance policies and procedures. The Quality Assurance Plan addresses the following types of activities.

- Establishing the applicability of published standards and procedures and determining the scope of the project standards and procedures.

- Monitoring the project and enforcement of compliance with all standards and procedures to facilitate the early detection of problems that could affect the reliability, maintainability, availability, integrity, safety, security, or usability of the product.

Work Product, continued:

- Inspecting hardware and software items and documenting for compliance to specifications and standards before their release to the test team or the system owner.

- Certifying deliverable items before their release to the system owner as compliant with all provisions of the project statement of work and contract, if applicable.

- Coordinating the project's technical problem reporting system and corrective action program to assure resolution of observed discrepancies.

- Measuring the quantitative and auditable progress of the project based on cost, schedule status, and quality status.

- Assuring consistent management and technical practices and the integrity of the product.

Provide enough information in the plan so that compliance can be monitored by means of project records. Whenever feasible, acquire automated tools to check compliance with project standards. For example, many CASE (computer-aided software engineering) tools can check compliance with standards, while checking the validity and consistency of requirements, design, and logic diagrams.

Review Process: Conduct a structured walkthrough to validate that the quality assurance policies and procedures are appropriate and adequate for the project.

Resources: The following resources are available from the Software Quality & Systems Engineering web site to assist with the development of a Quality Assurance Plan.

- Software Quality Assurance Plan Template
- Software Quality Assurance Plan Example
- Automated Office Systems Support Quality Assurance Model

The Institute of Electrical and Electronics Engineers Standard for Software Quality Assurance Plans (Std 730-1998) provides guidance on developing these plans.

Activity: **3.10 Develop Configuration Management Plan**

Responsibility: Project Manager or Configuration Manager

Description: Configuration management (CM) is a set of procedures used to control changes to the project during all stages of the project lifecycle. The goals of configuration management are to identify the configuration of the product (i.e., work products and their descriptions) at given points in time, to systematically control changes to the configuration, and to maintain the integrity and traceability of the configuration throughout the lifecycle.

Configuration management includes the following activities:

- A CM plan is prepared for each project according to a documented procedure and is coordinated with affected groups and individuals.
- A documented and approved configuration management plan is used as the basis for performing the configuration management activities.
- Authority for managing the project's baselines is established (e.g., a Configuration Control Board - CCB).
- A CM library system is established as a repository for the baselines.
- Work products are identified and placed under configuration control.
- Change requests and problem reports for all items/units are initiated, recorded, reviewed, approved, and tracked according to a documented procedure.
- Changes to baselines are controlled according to a documented procedure.
- Products from the baseline library are created and their release is controlled according to a documented procedure.
- Status of items/units is recorded according to a documented procedure.
- Standard reports documenting the CM activities and the contents of the baseline are developed and made available to affected groups and individuals.

$ Baseline audits are conducted according to a documented procedure.

Description, continued:

The work products placed under CM include the products that are delivered to the customer (e.g., the requirements document and the source code) and the items that are identified with or required to create these products (e.g., the compiler). A baseline library is established containing the baselines of the configuration items as they are developed. Changes to baselines and the release of products built from the baseline library are systematically controlled via the change control and configuration auditing functions of configuration management.

The Configuration Manager (or the individual assigned configuration responsibilities) is responsible for routine evaluation of the product. The Configuration Manager controls changes that are introduced into the systems product environment. The CM manager is responsible for the processes necessary to correct faults in the environment and product. The CM manager is not responsible for any overall project deadlines or management issues.

Work Product:

A CM plan that defines the configuration management policies and procedures is required for each project. The plan is developed early in the lifecycle to ensure the control of changes as soon as the project requirements are approved and baselined. In this stage, the plan addresses activities that are platform independent, such as identifying the items that will be placed under configuration management. As the project progresses through the lifecycle stages, the plan is expanded to reflect platform specific activities.

The CM plan addresses the following types of responsibilities and activities:

- Defining the required configuration management policy and procedures.
- Maintaining all documents in a central library where they can be accessed with ease.
- Receiving unit test and integrated builds and related documentation from the developer.

$ Performing version control procedures on unit and integrated builds received from the developer.

$ Packaging a tested unit or integrated build for return to developer or production as required.

$ Shipping an approved unit or integrated build to production as required.

$ Maintaining an archive of project-related correspondence between members of the project team.

Work Product, continued:

$ Overseeing the release and subsequent distribution of configuration items.

Provide enough information in the plan so that compliance can be monitored by means of project records. Whenever feasible, acquire automated CM tools to check compliance with project standards, the validity and consistency of product design, requirements, and system performance.

Based on the complexity of the project and the anticipated volume of changes, a Configuration Management Plan can be developed for a specific project, an existing plan can be modified to suit the requirements of a project, or a plan can be developed to manage all of the projects supporting a particular system owner's organization. Place a copy of the CM Plan in the Project File.

Review Process: Conduct structured walkthroughs to validate that the configuration management approach, the configuration identification, change control, status accounting, and auditing procedures are appropriate for the project.

Resources:

The Software Quality & Systems Engineering web site contains sample CM plans and a template.

The Institute of Electrical and Electronics Engineers Standard for Software Configuration Management Plans (Std 828-1990) provides guidance on developing these plans.

Chapter: **4.0 Requirements Definition Stage**

Description: The primary goal of this stage is to develop a basis of mutual understanding between the system owner/users and the project team about the requirements for the project. The result of this understanding is an approved Requirements Specification that becomes the initial baseline for product design and a reference for determining whether the completed product performs as the system owner requested and expected. All system requirements, (e.g., software, hardware, performance, functional, infrastructure, etc.) should be included.

This stage involves analysis of the system owner/users' business processes and needs, translation of those processes and needs into formal requirements, and planning the testing activities to validate the performance of the product.

Input: The following work products provide input to this stage.

- Project File
- Description of user environment
- Statement of project scope and objectives
- Statement of high-level project requirements
- Functional area contact list and project profile
- Summary of platform options
- Statement of project feasibility
- Analysis of Benefits and Costs Report
- Feasibility Study Document
- Project Plan
- Quality Assurance Plan
- Configuration Management Plan

High-Level Activities: The remainder of this chapter is divided into sections that describe specific high-level activities performed during this stage. These activities represent the minimum requirements for a large systems engineering effort. *Notes* are provided, as applicable, to assist in customizing these lifecycle stage requirements to accommodate different sizes of systems engineering efforts. The high-level activities are presented in the sections listed below.

4.1 Requirements Management
4.2 Select Requirements Analysis Technique
4.3 Define Project Requirements
4.4 Compile and Document Project Requirements
4.5 Establish Functional Baseline
4.6 Develop Project Test Plan
4.7 Develop Acceptance Test Plan
4.8 Select Design Technique

Output: Several work products are developed during this stage. The work products listed below are the minimum requirements for a large systems project. Deviations in the content and delivery of these work products are determined by the size and complexity of a project. Explanations of the work products are provided under the applicable activities described in the remainder of this chapter.

- Description of analysis technique
- Records of all project requirements
- User-oriented requirements manual *(optional)*
- Continuity of Operations Statement/Plan
- Data Dictionary
- Requirements Traceability Matrix
- Requirements Specification (draft)
- Project Test Plan
- Acceptance Test Plan *(draft)*
- Design technique
- Project Plan *(revised)*

A matrix showing the work products associated with each high-level activity is provided in *Exhibit 4.0-1, Requirements Definition Stage Activities and Work Products by Project Size.* The matrix also shows which work products are deliverables and whether they are required or optional for small, medium, and large products.

Once the requirements are baselined changes must be managed through the established change process, which may include a change control board.

Review Process: Quality reviews are necessary during this stage to validate the product and associated work products. The activities that are appropriate for quality reviews are identified in this chapter and Chapter 2.0, Lifecycle Model.

Structured Walkthrough
Requirements for a peer review or a more formal structured walkthrough, are documented under *Review Process*, at the end of each Task, Subtask, or Activity section in this stage.

In-Stage Assessment
Schedule at least one ISA prior to the Requirements Definition Stage Exit process. Additional ISAs can be performed during the stage, as appropriate. An ISA is recommended after the completion of the Requirements Specification.

Stage Exit
Schedule a Stage Exit as the last activity of the Requirements Definition Stage to enable the project approvers to review project deliverables and provide a concur/non-concur position to the project manager.

References: Chapter 2.0, Lifecycle Model, Q*uality Review*, provides an overview of the Quality Reviews to be conducted on a project.

Conducting Structured Walkthroughs provides a procedure and sample forms that can be used for structured walkthroughs.

In-Stage Assessment Process Guide provides a procedure and sample report form that can be used for in-stage assessments.

Stage Exit Process Guide provides a procedure and sample report form that can be used for stage exits.

Resource: DOE Software Quality & Systems Engineering web site.

Bibliography: The following materials were used in the preparation of the Requirements Definition Stage chapter.

1. Carnegie Mellon University, Software Engineering Institute, *Capability Maturity Model: Guidelines for Improving the Software Process,* Addison Wesley Longman, Inc., 1994.

2. *Software Engineering Handbook*, Chapter 4, Software Requirements Analysis.

3. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Guide to Software Requirements Specifications,* ANSI/IEEE Std 830-1998, New York, 1998.

4. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Lifecycle Processes,* IEEE Std 1074-1991, New York, 1992.

5. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Software Verification and Validation Plans,* ANSI/IEEE Std 1012-1998, New York, 1998.

6. U.S. Department of Energy, Nuclear Weapons Complex , Quality Managers, Software Quality Assurance Subcommittee, *Guidelines for Software Requirements Management,* July 1998.

Exhibit 4.0-1. Requirements Definition Stage Activities and Work Products by Project Size

Work Activity		Project Size L	M	S	Work Product	Deliverables L	M	S
4.1	Requirements Management	R	R	R	Requirements Traceability Matrix Change Request Form Change Control Log	R A A	R A A	A A A
4.2	Select Requirements Analysis Technique	R	R	R	Description of Analysis Technique	I	I	I
4.3	Define Project Requirements	R	R	R	Records of Project Requirements User-Oriented Requirements Manual *(optional)* Continuity of Operations Statement/Plan Data Dictionary	I O R R	I O R R	I O R R
4.4	Compile and Document Project Requirements	R	R	R	Requirements Specification *(draft)*	R	R	R
4.5	Establish Functional Baseline	R	R	R	Requirements Specification *(final)*	R	R	R
4.6	Develop Project Test Plan	R	R	A	Project Test Plan	R	R	A
4.7	Develop Acceptance Test Plan	R	R	R	Acceptance Test Plan *(draft)*	R	R	R
4.8	Select Design Technique	R[2]	R[2]	R[2]	Design Technique	N[2]	N[2]	N[2]
3.8	Revise Project Plan	R	R	R	Project Plan *(revised)*	R	R	R
2.5	Conduct Structured Walkthrough(s)	R	R	A	Structured Walkthrough Management Summary Report	N	N	N
2.5	Conduct In-Stage Assessment	R	R	A	ISA Report Form[1]	N	N	N
2.5	Conduct Requirements Definition Stage Exit	R	R	A	Stage Exit Meeting Summary	N	N	N

Size: L = Large, M = Medium, S = Small

Minimum Requirements: R = Required, A = As Appropriate, N = Not a Deliverable[3], I = Input to another Deliverable

O = Optional work product

[1] = Completed by reviewer

[2] = Can adapt an existing plan

[3]A deliverable is a work product that is identified in the Project Plan and is formally delivered to the system owner and other project stakeholders for review and approval.

Activity: **4.1 Requirements Management**

Responsibility: Project Manager

Description: Requirements management is essentially a program composed of gathering, organizing, prioritizing, and documenting requirements; verifying that requirements have been captured in the product, and managing changes to requirements.

Gathering, organizing, prioritizing and documenting requirements is an interactive communication process and working relationship between stakeholders and the project team to discover, define, refine, and record a precise representation of the product requirements.

Requirements management documents the needs, expectations, and understanding of the product to be delivered and provides a framework for identifying, planning, scheduling, costing, verifying, tracing, testing, evaluating, changing, and renegotiating requirements to satisfy stakeholder needs and expectations of the project. When requirements are initially gathered, some or all will be planned for the current project (e.g., initial release). The requirements for the project are documented in the Requirements Specification. As the project progresses, more requirements may be identified and managed through a change control process. As part of requirements management, the project manager must track requirements that are accepted for the current project and those that will be planned for subsequent releases.

Each requirement identified in the Requirements Specification document should be uniquely identified in a Requirements Traceability Matrix. The Requirements Traceability Matrix is a requirements management tool that ensures requirements are traced and verified through the various lifecycle stages; especially design, testing, and implementation. Requirements must be traceable from external sources such as the customer, to derived system-level requirements, to specific hardware/software product requirements.

Work Products: A substantial amount of information that is used for requirements management in later stages in the systems engineering process is gathered in the Requirements Definition Stage. The Requirements Traceability Matrix is a work product that is created during the Requirements Definition Stage and used to verify and validate that requirements are met and the product remains within scope. Refer to each task for information on applicable work products.

Resources: Templates for the Requirements Traceability Matrix and Requirements Specification document are provided on the Software Quality & Systems Engineering web site.

Templates for the Software Change Request Form and the Software Change Control Log are provided on the Software Quality & Systems Engineering web site.

Carnegie Mellon University, Software Engineering Institute, *Capability Maturity Model: Guidelines for Improving the Software Process*, Addison Wesley Longman, Inc., 1994.

Tasks: The following tasks are involved in requirements management.

4.1.1 Develop Requirements Traceability Matrix
4.1.2 Implement Requirements Change Control

Task: **4.1.1 Develop Requirements Traceability Matrix**

Description: A requirements traceability matrix is a tool used to trace project lifecycle activities and work products to the project requirements. The matrix establishes a thread that traces requirements from identification through implementation.

Every project requirement must be traceable back to a specific project objective(s) described in the Project Plan. This traceability assures that the product will meet all of the project objectives and will not include inappropriate or extraneous functionality.

All work products developed during the design, code, and testing processes in subsequent lifecycle stages must be traced back to the project requirements described in the Requirements Specification. This traceability assures that the product will satisfy all of the requirements and remain within the project scope.

It is also important to know the source of each requirement, so that the requirements can be verified as necessary, accurate, and complete. Meeting conference records, user survey responses, and business documents are typical sources for project requirements.

Work Product: Develop a matrix to trace the requirements back to the project objectives identified in the Project Plan and forward through the remainder of the project lifecycle stages. Place a copy of the matrix in the Project File. Expand the matrix in each stage to show traceability of work products to the requirements and vice versa. The requirements traceability matrix should contain the following fields:

- A unique identification number containing the general category of the requirement (e.g., SYSADM) and a number assigned in ascending order (e.g., 1.0; 1.1; 1.2).
- The requirement statement.
- Requirement source (Conference; Configuration Control Board; Task Assignment or other).
- Requirements Specification and/or Functional Requirements Document paragraph number containing the requirement.
- Design Specification paragraph number containing the requirement.

Work Product, continued

- $ Program Module containing the requirement.
- $ Test Specification containing the requirement test.
- $ Test Case number(s) where requirement is to be tested (optional).
- $ Verification of successful testing of requirements.
- $ Modification field. If requirement was changed, eliminated, or replaced, indicate disposition and authority for modification.
- $ Remarks.

Review Process: Conduct a structured walkthrough of the Requirements Traceability Matrix to ensure that all requirements have been accurately captured.

Sample Traceability Matrix: One method for tracing requirements is a threading matrix that groups requirements by project objectives. Under each project objective, the source of the requirement, the unique requirement identification number, and the lifecycle activities are listed in columns along the top and the project requirements in rows along the left side. As the project progresses through the lifecycle stages, a reference to each requirement is entered in the cell corresponding to the appropriate lifecycle activity. *Exhibit 4.1-1, Sample Requirements Traceability Matrix*, provides an example.

Resource: A template for the Requirements Traceability Matrix is provided on the Software Quality & Systems Engineering web site.

Exhibit 4.1-1. Sample Requirements Traceability Matrix

Requirement	Source	Unique Number	Reqts Spec/ Functional Requirement Document	Design Spec.	Program Module	Test Spec.	Test Case(s)	Successful Test Verification	Modifica-tion of Reqt	Remarks
Objective 1: Security										
The product shall have three user access levels with the capability to add new access levels in the future.	conference record dated 5/19/02	SYSADM 1.0								
Each user access level shall have a unique designation.	conference record dated 5/19/02	SYSADM 1.1								
One user access level shall allow read-only access to the production database.	conference record dated 5/19/02	SYSADM 1.2								
The second user access level shall allow read and write access to the production database.	conference record dated 5/19/02	SYSADM 1.3								
The third user access level shall allow read, write, and delete access to all application databases.	conference record dated 5/19/02	SYSADM 1.4								

Task: **4.1.2 Implement Requirements Change Control**

Description: As a project progresses, more requirements may be identified. Using the change control process, the project manager tracks requirements that are accepted for the current project and those that are planned for subsequent releases.

Changes to requirements are initiated via a formal change request form. They are then logged and tracked according to the change management process to ensure the changes are included in the traceability matrix and testing and acceptance plans. Other work products are revised as appropriate.

When a change becomes known, the person initiating the change should document the request on a change request form. The change request form captures as much detail about the requirement as possible, (e.g., its effect on the system, procedures and documentation) as well as the reason and priority of the change. A separate change request form should be completed for each requested change.

When the responsible person receives the formal change request form, the change should be recorded on a change control log. Once logged, the request should be provided to the development staff for evaluation as to scope of effort.

Once the evaluation is completed, the project stakeholders and project manager should evaluate the impact to the project and the priority level of the change request to determine whether or not to approve the change for inclusion in the current project or deferral for a future project. Approvals should be recorded on the change request form.

The status of the change request should be managed through on the change control log and updated as the status changes.

Once a change is approved, the requirements traceability matrix and all other appropriate work products, (e.g, test plans or acceptance plans) are updated to include the new requirement. If scheduling is impacted by the change, the Project Plan should is updated.

Work Product: As changes to the requirements are requested, the completed Change Request Forms and a copy of the Change Control Log should be maintained in the Project File.

Review Process: A peer review or structured walkthrough may be conducted on the Change Request Forms and Change Control Log.

Resource: Templates for the Change Request Form and the Change Control Log can be found on the Software Quality & Systems Engineering web site.

Activity: **4.2 Select Requirements Analysis Technique**

Responsibility: Project Manager/Team

Description: A requirements analysis technique is the set of data collection and analysis techniques (e.g., user interviews and rapid prototyping) combined with the lifecycle requirements standards (e.g., tracing the requirements through all lifecycle activities) that are used to identify the project requirements and to define exactly what the product must do to meet the system owner/users' needs and expectations. When appropriate, the technique must include methods for collecting data about users at more than one geographic location and with different levels and types of needs.

The requirements analysis technique should be in harmony with the type, size, and scope of the project; the number, location, and technical expertise of the users; and the anticipated level of involvement of the users in the data collection and analysis processes. The technique should ensure that the functionality, performance expectations, and constraints of the project are accurately identified from the system owner/users' perspective. The technique should facilitate the analysis of requirements for their potential impact on existing operations and business practices, future maintenance activities, and the ability to support the system owner's long-range information resource management plans.

It is advantageous to select a technique that can be repeated for similar projects. This allows the project team and the system owner/users to become familiar and comfortable with the technique.

Discuss the analysis technique with the system owner and users to make sure they understand the process being used, their role and responsibilities in the process, and the expected format of the output (e.g., how the requirements will be organized and described).

Work Product: Create a description of the analysis technique and share it with all members of the project team, system owner, and users. Place a copy of the analysis technique description in the Project File.

Review Process: Conduct a structured walkthrough to verify that the requirements analysis technique is appropriate for the scope and objectives of the project. A structured walkthrough is not needed when the technique has been used successfully on similar projects for the same system owner/user environment.

Activity: **4.3 Define Project Requirements**

Responsibility: Project Manager/Team

Description: Use the project scope, objectives, and high-level requirements as the basis for defining the project requirements. The questions used to define the project objectives may be helpful in developing the project requirements. The goals for defining project requirements are to identify what functions are to be performed on what data, to produce what results, at what location, and for whom.

The requirements must focus on the products that are needed and the functions that are to be performed. Avoid incorporating design issues and specifications in the requirements. One of the most difficult tasks is to determine the difference between "what" is required and "how to" accomplish what is required. Generally, a requirement specifies an externally visible function or attribute of a system (i.e., "what".) A design describes a particular instance of how that visible function or attribute can be achieved (i.e., "how to".)

Requirements should be specified as completely and thoroughly as possible. The requirements must support the system owner's business needs, information resource management long-range plans, and the organizational and Departmental missions. When requirements are being defined, it is not sufficient to state only the requirements for the problems that will be solved; all of the requirements for the project must be captured.

Attributes: Each requirement must be stated as a unique objective with the following attributes. The existence of these attributes must be verified prior to the delivery of the Requirements Specification later in the Requirements Definition Stage.

- Necessary - Absolute requirements that are to be verified are identified by "must" or "shall". Goals or intended functionality are indicated by "will".
- Correct - Each requirement is an accurate description of a feature or process of the product.
- Unambiguous - The statement of each requirement denotes only one interpretation.
- Complete - Each requirement describes one result that must be achieved by the product. The requirement should not describe the means of obtaining the result.

Attributes, continued:

- $ Consistent - Individual requirements are not in conflict with other requirements.
- $ Verifiable (testable) - Each requirement is stated in concrete terms and measurable quantities. A process should exist to validate that the product (when developed) will satisfy the set of requirements.
- $ Modifiable - The structure and style of the requirements are such that any necessary changes to the requirements can be made easily, completely, and consistently.
- $ Traceable - The origin of each requirement is clear and can be tracked in future development activities and tests.

Identification System: The creation of a standard identification system for all requirements is required in order to facilitate configuration control, requirements traceability, and testing activities. The identification system must provide a unique designator for each requirement. For example, the identification system can classify the requirements by type (e.g., functional, input, or computer security). Within each type classification, the requirements can be assigned a sequential number. Select an identification system that is appropriate for the scope of the project.

Changes: As the project evolves, the requirements may change or expand to reflect modifications in the users' business plans, design considerations and constraints, advances in technology, and increased insight into user business processes. A formal change control process must be used to identify, control, track, and report proposed and approved changes. Approved changes in the requirements must be incorporated into the Requirements Specification in such a way as to provide an accurate and complete audit trail of the changes. This change control process should be an integral part of the project's Configuration Management Plan.

Tasks: The following tasks are involved in defining project requirements.

4.3.1 Define Functional Requirements
4.3.2 Define Input and Output Requirements
4.3.3 Define Performance Requirements

Tasks, continued:

4.3.4	Define User Interface Requirements
4.3.5	Define System Interface Requirements
4.3.6	Define Communication Requirements
4.3.7	Define Computer Security and Access Requirements
4.3.8	Define Backup and Recovery Requirements
4.3.9	Define Data Requirements
4.3.10	Define Implementation Requirements

Task: **4.3.1 Define Functional Requirements**

Description: Functional requirements define what the product must do to support the system owner's business functions and objectives. The functional requirements should answer the following questions.

- How are inputs transformed into outputs?
- Who initiates and receives specific information?
- What information must be available for each function to be performed?

Identify requirements for all functions whether they are to be automated or manual. Describe the automated and manual inputs, processing, outputs, and conditions for all functions. Include a description of the standard data tables and data or records that will be shared with other applications. Identify the forms, reports, source documents, and inputs/outputs that the product will process or produce to help define the functional requirements.

Develop a functional model to depict each process that needs to be included. The goal of the functional model is to represent a complete top-down picture of the product.

Use flow diagrams to provide a hierarchical and sequential view of the system owner's business functions and the flow of information through the processes.

Work Product: Maintain a record of all functional requirements. Save for incorporation into the Requirements Specification. Place a copy of the functional requirements in the Project File.

Optional Work Product: Consider developing an optional work product that defines how the final product will operate to support the system owner organization's business functions and objectives. This user-oriented requirements manual would identify processes in a narrative form from the user's perspective and would include requirements for all functions whether they are to be automated or manual. A functional description can be developed to depict each process that will be provided. The goal is to present a complete top-down picture of the product.

Optional Work Product, continued: This user-oriented requirements manual can be used as an aid in validating the functional requirements and serves as the basis for the user documentation. If a test group outside the project team is used, the test group can work with the project team to develop the manual.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability, accuracy, and completeness of the functional requirements.

Task: **4.3.2 Define Input and Output Requirements**

Description: Describe all manual and automated input requirements for the product such as data entry from source documents and data extracts from other applications; include where the inputs are obtained.

Describe all output requirements for the product such as printed reports, display screens, and files; include who or what is to receive the output.

Work Product: Maintain a record of all input and output requirements. Save for incorporation into the Requirements Specification. Place a copy of the input and output requirements in the Project File.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability, accuracy, and completeness of the input and output requirements.

Task: **4.3.3 Define Performance Requirements**

Description: Performance requirements define how the product must function (e.g., hours of operation, response times, and throughput under various load conditions). The information gathered in defining the project objectives can translate into very specific performance requirements; (e.g., if work performed for an organization is mission essential to the Department, the hours of operation and throughput will be critical to meeting the mission). Also, Government and DOE policy can dictate specific availability and response times.

Work Product: Maintain a record of all performance requirements. Save for incorporation into the Requirements Specification. Place a copy of the performance requirements in the Project File.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability, accuracy, and completeness of the performance requirements.

Task: **4.3.4 Define User Interface Requirements**

Description: The user interface requirements should describe how the user will access and interact with the product, and how information will flow between the user and the product.

Interface Issues: A standard set of user interface requirements may be established for the system owner organization. If not, work with the system owner and users to develop a set of user interface requirements that can be used for all automated products for the system owner's organization. A standard set of user interface requirements will simplify the design and code processes, and ensure that all automated products have a similar look and feel to the users. When other constraints (such as a required interface with another application) do not permit the use of existing user interface standards, an attempt should be made to keep the user interface requirements as close as possible to the existing standard.

The following are some of the issues that should be considered when trying to identify user interface requirements.

- The users' requirements for screen elements, navigation, and help information.
- The standards issued by the Federal Government, DOE organization, and industry that apply to user interfaces.
- The classification of the users who will access and use the product.
- The range of work that the users will be performing with the product.

Define the user interface requirements by identifying and understanding what is most important to the user, not what is most convenient for the project team.

Work Product: Maintain a record of all user interface requirements. Save for incorporation into the Requirements Specification. Place a copy of the user interface requirements in the Project File.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability, accuracy, and completeness of the user interface requirements.

Task: **4.3.5 Define System Interface Requirements**

Description: The hardware and software interface requirements must specify hardware and software interfaces required to support the development, operation, and maintenance of the product.

The following information should be considered when defining the hardware and software interface requirements.

- System owner's and users' IT environment.
- Existing or planned software that will provide data to or accept data from the product.
- Other organizations or users having or needing access to the product.
- Purpose or mission of interfacing software.
- Common users, data elements, reports, and sources for forms/events/outputs.
- Timing considerations that will influence sharing of data, direction of data exchange, and security constraints.
- Development constraints such as the operating system, database management system, language compiler, tools, utilities, and network protocol drivers.
- Standardized system architecture defined by hardware and software configurations for the affected organizations, programmatic offices, sites, or telecommunications operations.

Work Product: Maintain a record of all system interface requirements. Save for incorporation into the Requirements Specification. Place a copy of the system interface requirements in the Project File.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability, accuracy, and completeness of the system interface requirements.

Task: **4.3.6 Define Communication Requirements**

Description: The communication requirements define connectivity and access requirements within and between user locations and between other groups and applications.

The following factors should be considered when defining communication requirements.

- Communication needs of the user and customer organizations.
- User organization's existing and planned communications environment (e.g., telecommunications; LANs, WANs, and dial-up).
- Projected changes to the current communication architecture, such as the connection of additional local and remote sites.
- Limitations placed on communications by existing hardware and software including:
  - user systems
  - applications that will interface with the product
  - organizations that will interface with the product
- Organization, Government, and industry standards that define communication requirements and limitations.
- Future changes that may occur during the project.

Work Product: Maintain a record of all communication requirements. Save for incorporation into the Requirements Specification. Place a copy of the communication requirements in the Project File.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability accuracy, and completeness of the communications requirements.

Task: **4.3.7 Define Computer Security and Access Requirements**

Description: Develop the computer security requirements in conjunction with the system owner's Computer System Security Officer and other stakeholders who provide competent input in the information system security area. This involvement affords early determination of classifications and levels of access protection required for the product.

If a product under development processes sensitive personal information, appropriate safeguards must be established to protect the information from accidental disclosure. Refer to the Office of Management and Budget web site for guidance on the Privacy Act.

Implement applicable security procedures to assure data integrity and protection from unauthorized disclosure, particularly during development efforts. The organization that owns the data defines the data classification. The project team must be aware of all the types of data and of any classified or proprietary algorithms used in the product.

Procedure: Use the following procedure to determine computer security requirements.

1. Identify the types of data that will be processed by the product.

2. Determine preliminary data protection requirements.

    a. For products processing classified information refer to DOE Manual 471.2-2, Classified Information Systems Security Manual.

    b. For products processing unclassified and unclassified sensitive information, refer to DOE Notice 205.1, Unclassified Cyber Security Program.

    c. For products processing sensitive personal information, contact the Freedom of Information Office for coordination and assistance in complying with policy and guidance.

    d. For products that are considered to be mission essential, refer to Notice 205.1 Unclassified Cyber Security Program and other relevant DOE Security Standards.

3. Coordinate with the owner of the host platform to identify existing supporting computer security controls, if applicable.

4. Incorporate security requirements into the Requirements Specification.

Sample Access Control Questions: The following list provides sample questions that can be used to help define the access controls for the product.

- What access restrictions are placed on the users by their organization or programmatic office?
- What are the audit and other checking needs for the product?
- What separation of duties, supervisory functions related to control, operating environment requirements, or other functions will impact the product?
- What measures will be used to monitor and maintain the integrity of the product and the data from the user's viewpoint?

Work Product: Maintain a record of all security and access requirements. Save for incorporation into the Requirements Specification. Place a copy of the security and access requirements in the Project File.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability, accuracy, and completeness of the computer security and access requirements.

Resources:

- DOE Notice 205.1, Unclassified Cyber Security Program, provides guidance for organizations to implement a computer security program for sensitive information.
- DOE Manual 471.2-2, Classified Information Systems Security Manual, provides guidance for classified systems.
- DOE 1800.1, PRIVACY ACT.
- DOE Order 471.1, Unclassified Controlled Nuclear Information.
- DOE Order 471.2A, Information Security Program.

Task: **4.3.8 Define Backup and Recovery Requirements**

Description: Develop the requirements for data backup, recovery, and operation startup for the product in conjunction with the site authority for continuity of operations. A checklist is provided in *Exhibit 4.3-1, Checklist for Identifying Mission-Essential Systems,* to determine if the product is mission essential. Additionally, ensure that the mission essential system is included in the Continuity of Operations or Disaster Recovery Plans.

Work Product: If a product is determined to be mission essential, a Continuity of Operations Plan must be developed. If the product is not mission essential, a Continuity of Operations Statement is required. Place a copy of the Continuity of Operations Statement or Plan in the Project File.

Review Process: Conduct structured walkthroughs as needed to assure the necessity, testability, accuracy, and completeness of the backup and recovery requirements.

Resource: *Disaster Recovery Program Guidelines;* Department of Energy; Office of Information Resource Management; Policy, Plans, and Oversight, July 1991.

A template for the Continuity of Operations Plan is available on the Software Quality & Systems Engineering web site.

Exhibit 4.3-1. Checklist for Identifying Mission-Essential System

The checklist is intended to help identify products that are mission essential. If a "yes" answer is selected for one or more of the criteria, the product is mission essential and a Continuity of Operations Plan must be developed.

	Criterion	Yes	No
1	Inability to perform function adversely affects national security.		
2	Inability to perform function adversely affects safety of individuals.		
3	Needed for military effort and civil defense activities during a national emergency.		
4	Needed for mobilization and protection of material and manpower during national emergency.		
5	Function required for maintenance of public health, safety, and order.		
6	Maintains records essential to preservation of legal rights.		
7	Large financial loss incurred with inability to perform functions.		
8	Large expense incurred if performing function by other means.		
9	Primary repository of information reported to Congress or other agencies.		
10	Critical for compliance with federal regulatory requirements.		
11	Sole source of data unobtainable by other means, or not easily recreated.		

Task: **4.3.9 Define Data Requirements**

Description: Data requirements identify the data elements and logical data groupings that will be stored and processed by the product. The identification and grouping of data begins during the Requirements Definition Stage and is expanded in subsequent stages as more information about the data is known.

Work Products: The major output of the data requirements identification process is a data dictionary. A data dictionary provides an ordered set of definitions about data inputs and outputs, and data stores. In the Requirements Definition Stage, the data dictionary contains a minimum amount of information about data elements such as definitions of the entities, how the data are stored, and data flows to or from other applications. The data dictionary is refined during the design stages as data elements are documented in more detail, and the logical groupings of data elements are formed into interrelated tables or record descriptions.

Maintain a record of all data requirements. Save for incorporation into the Requirements Specification. Place a copy of the data requirements in the Project File.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability, accuracy, and completeness of the data requirements.

Resource: The following is available guidance on electronic records management.

National Archives and Records Administration (NARA):

- Evaluating Electronic Record Keeping, A self-Inspection Guide for Agencies, November 1990.
- Managing Electronic Records, 1990.
- Agency Record Keeping Requirements: A Management Guide, 1995.
- NARA NA Form 14028, Information Systems Descriptions.

DOEF1324.10, Records Inventory and Disposition Schedule (RIDS).

DOE Records Management web site.

GSA, Electronic Record Keeping, July 1989.

Task: **4.3.10 Define Implementation Requirements**

Description: Describe the requirements anticipated for implementing the product (e.g., user production cycle). The high-level implementation requirements are identified early in the lifecycle to support decisions that need to be made for the information systems engineering approach. The implementation requirements are expanded into a full implementation approach during the design stages.

The following paragraphs provide highlights of some of the implementation requirements that need to be considered.

Operating Environment: Identify any capacity restrictions on the existing hardware or software that needs to be addressed and identify any hardware or software that needs to be acquired (e.g., communication hardware, file servers, off-the-shelf software, network interface cards, and LAN utilities).

Acquisition: If hardware or software must be acquired, identify the necessary acquisition activities. These activities include preparing specifications, estimating costs, scheduling procurement activities, selection, installation, and testing.

Conversion: Identify requirements for converting data from an existing or external application to the new product. Consider requirements for data entry, data protection, computer time, conversion programs, personnel, and other resources that will be needed. Also identify the requirements for the conversion of software, if necessary. Implementing a new application may involve converting software from one environment to another, or modifying software to interface with other applications. Include requirements for testing the conversion process and validating that it was successfully accomplished.

Installation: Identify the installation requirements for any new hardware, operating system, or software. For hardware installations, consider environmental factors such as air conditioning, power supply, and security requirements. For software installations, consider proprietary software such as database management systems. For application software, consider the installation of the application's programs, parallel operation of the old and new applications, or the cutover from a test to a production environment. Hardware and software installation must be coordinated with the work cycles of the user organization to create a minimum of disruption, and to assure that data are available as needed. Installation must be scheduled to assure that, when data conversion is necessary, the needed data are protected.

Training: Identify the specific training needs for various categories of users and administrators. Also identify training requirements for personnel time, computer time, training facilities, and training database(s).

Documentation: Identify requirements for the development and distribution of operational documentation for support personnel and user documentation. Operational documentation may include job control procedures and listings, operational instructions, system administration responsibilities, archiving procedures, and error recovery. User documentation includes the users manual, step-by-step instructions, online documentation, and online help facilities.

Work Product: Maintain a record of all implementation requirements. Save for incorporation into the Requirements Specification. Place a copy of the implementation requirements in the Project File. This information will also be used to develop an Implementation Plan in the Functional Design Stage.

Review Process: Conduct structured walkthroughs as needed to ensure the necessity, testability, accuracy, and completeness of the implementation requirements.

Activity: **4.4 Compile and Document Project Requirements**

Responsibility: Project Manager/Team

Description: Compile the requirements gathered during the requirements analysis process in preparation for the development and delivery of the draft Requirements Specifications. The following steps should be performed as part of the requirements compilation activity.

- Select and use a standard format for describing the requirements. Ensure compliance with Information Architecture and any site specific standards.
- Present the logical and physical requirements without dictating a physical design or technical solutions.
- Write the requirements in nontechnical language that can be fully understood by the system owner and users.
- Organize the requirements into meaningful groupings (e.g., all security-related requirements or all requirements for generating reports).
- Develop a numbering scheme for the unique identification of each requirement.
- Select a method for: (1) tracing the requirements back to the sources of information used in deriving the requirements (e.g., specific system owner/user project objectives); and (2) threading requirements through all subsequent lifecycle activities (e.g., testing).

Resource: A template for a Requirements Specification is provided on the Software Quality & Systems Engineering web site.

Task: The following task is involved in the compilation of the project requirements.

4.4.1 Develop Requirements Specification

Task: **4.4.1 Develop Requirements Specification**

Description: The Requirements Specification describes the inputs to be supplied by the user or other sources, the processing that needs to occur, and the outputs desired by the user or required by interfacing systems. The emphasis should be placed on specifying product functions without implying how the product will provide those functions. This approach provides maximum flexibility for the product designers. The how-to of product implementation is determined in the design stages.

Work Product: Prepare the Requirements Specification by integrating all of the requirements developed during this stage. Several formats are available for organizing the requirements information (e.g., from a functional perspective or a data processing perspective).

Document all design constraints including processing, performance, interface, resource, safety, security and reliability requirements. Document any assumptions made. Define data constraints such as limits, formats, messages, commands, and displays.

Review Process: Conduct structured walkthroughs as needed to ensure that the Requirements Specification is accurate, complete, and expresses the requirements in a manner that can be understood by the system owner.

The completion of the draft Requirements Specification is an appropriate time to schedule an In-Stage Assessment (ISA).

Reference: The *In-Stage Assessment Process Guide* provides a description and instructions for conducting an ISA.

Resource: A template of the Requirements Specification document is available on the Software Quality & Systems Engineering web site.

Activity: **4.5 Establish Functional Baseline**

Responsibility: Project Manager/Team

Description: The functional baseline, sometimes called a system requirements baseline, is the main technical work product of the Requirements Definition Stage. The system requirements are baselined after the system owner's formal approval of the Requirements Specification. Once the requirements are baselined, any changes to the requirements must be managed under change control procedures established in the Configuration Management Plan. Approved changes must be incorporated into the Requirements Specification.

Work Product: Prepare the final Requirements Specification and submit to the system owner and users for their review and approval. The approved Requirements Specification is the official agreement and authorization to use the requirements for the product design. Approval implies that the requirements are understood, complete, accurate, and ready to be used as the basis for the subsequent lifecycle stages.

It is important for the system owner/users to understand that changes to the approved Requirements Specification affect the project scope and therefore can change the project cost, resources, or schedule. It is the responsibility of the project manager and project team to identify system owner/user requested changes that would result in a change of project scope; evaluate the potential impact to the project costs, resources, or schedule; and notify the system owner of the project planning revisions that will be required to accommodate their change requests.

Place a copy of the approved Requirements Specification in the Project File.

Review Process: The Requirements Specification should be reviewed by the system owner and users. After making the changes needed to resolve problems found during the review, the functional baseline is formally established upon receipt of the system owner's approval.

Activity: **4.6 Develop Project Test Plan**

Responsibility: Project Manager

Description: The Project Test Plan is a narrative and tabular description of the test activities planned for the project during development or enhancement. The Project Test Plan should establish the testing necessary to validate that the project requirements have been met and that the deliverables are at an acceptable level in accordance with existing standards. The plan also ensures that a systematic approach to testing is established and that the testing is adequate to verify the functionality of the product.

The Project Test Plan includes the resources, project team responsibilities, and management techniques needed to plan, develop, and implement the testing activities that will occur throughout the lifecycle. If individuals outside of the project team perform system and acceptance testing, the plan includes the responsibilities and relationships of external test groups.

In this stage, the plan is written at a high level and focuses on identifying test techniques and test phases. Detailed information about test products (i.e., test plans, test procedures, and test reports) is added to the Project Test Plan as the project progresses through subsequent lifecycle stages.

Development of the Project Test Plan is the responsibility of the project manager. If a test group outside the project team will be involved in any test phase, the project manager must coordinate the Project Test Plan with each test group.

The Project Test Plan must be reviewed and approved by the system owner prior to conducting any tests.

Note: For small projects, a formal Project Test Plan may not be necessary; however, a test approach and testing are required and must be documented.

Note: Typically, the project Test Plan covers all test phases including unit, integration, system, and acceptance. For large or risky projects, a separate Acceptance Test Plan may be required (see section 4.7 for a description of the Acceptance Test Plan).

Work Product: When the Project Test Plan is complete, it should contain the following information:

- Describe the occurrence and timing of the test phases in the lifecycle and the entrance and exit criteria for each test phase.

Work Product, continued:

- Specify the test products at each test phase. Describe the types and scope of the testing activities to be performed on each component of the application and the group who is responsible to produce them.
- Map what requirements are verified in what test phase.
- Establish the criteria for evaluating the test results of each test phase.
- Make an initial determination of the resources necessary to accomplish the testing.
- Identify the appropriate person or group to conduct each type of testing activity.
- Outline the test environment (hardware, software, test tools, and data) needed to conduct the tests.
- Develop a preliminary schedule for executing the test activities.

Place a copy of the Project Test Plan in the Project File.

Review Process: Conduct structured walkthroughs to assure the Project Test Plan document adequately describes all testing activities, test schedules, test products, test responsibilities, the testing methodology, and the required resources.

Resources: Templates for Project Test Plans and Acceptance Test Plans are provided on the Software Quality & Systems Engineering web site.

Tasks: Preparation of the Project Test Plan involves the following tasks.

4.6.1 Identify Test Techniques
4.6.2 Identify Test Phases
4.6.3 Identify Test Environment Requirements

Task: **4.6.1 Identify Test Techniques**

Description: The Project Test Plan should specify the testing techniques planned for the project including the types of tests required, test documents, test methods, and test data collection. Each test from unit through acceptance testing is specified in terms of entrance and exit criteria and the expected level of involvement from the project team, test group, and other functional areas.

Unit and integration tests with appropriate data must be developed to exercise and validate all specified application requirements, functions, and objectives. System and acceptance tests validate that the integrated system meets the requirements.

Each type of test must use controlled computer generated or live data as specified. The test data must be prepared to include values that will verify the functional capabilities of the test component, identify its limitations and deficiencies (if any), exercise its capabilities, and verify that the component performs its intended function as required.

If pilot testing or a phased implementation is required for the product, the Project Test Plan should include such requirements. In the case of an implementation involving phased releases, the plan should include the requirements for regression testing of the complete application as new elements are introduced.

For each type of test conducted, the test results are compared with the expected results. Discrepancies are identified and any problems resolved. Retesting is required to verify that the problem solution eliminates the problem and does not introduce new errors. The final test results are accompanied by a completed test results/error log form. This form is completed by the individual(s) responsible for testing and attached to the documents that certify the completion of each type of test.

Task: **4.6.2 Identify Test Phases**

Description: The product should be tested in four sequential phases: unit, integration, system, and acceptance. Some projects may require additional types of tests (such as prototype testing for offsite installations). The four test phases and prototype testing are described below.

Unit Test Phase: The unit test phase involves testing of the individual units or groups of related units. A unit is a component that is not subdivided into other components; it is a logically separable part of a computer program. Evaluate each unit of code on how well it meets the performance requirements for which it was designed. Consider timing, memory, accuracy in producing numerical and logical results; and the preparation of input and output required for validating program logic, syntax, and performance requirements. This test phase is performed by the developers responsible for writing the code.

Integration Test Phase: Integration testing is an orderly progression of testing in which software elements, hardware elements, or both are combined and tested to evaluate the interaction between them. Each program/module must be tested. Integration testing is required to validate that groups of related programs, when combined to establish an integrated functional module of code, interface properly, and perform the functions for which they were designed. Examine the source program/module statements to ensure that the program logic meets the requirements of the design and that the application satisfies an explicit functional requirement. This test phase is performed by the project team.

System Test Phase: The system test phase tests the integrated hardware and software to verify that the product meets its specified requirements and operates successfully on the host platform. This test phase is required to validate, when the entire product is loaded onto the host platform, that the proper initialization is performed; decision branching paths are appropriate; and all functions are performed as specified in the Requirements Specification. System testing validates that the product produces the required outputs and interfaces properly with other systems with which the product gives or receives data; that transaction response times meet user expectations; and machine resource allocation and utilization are within expected norms. This test phase can be performed by the project team or by an independent test group with support from the project team.

Acceptance Test Phase: Acceptance testing is conducted to determine whether a product satisfies its acceptance criteria and to enable the system owner's organization to determine whether to accept the product. The acceptance test is required to validate that the system, its related documentation and tools, satisfy all of the specified requirements and objectives of the system owner's organization, DOE standards, the requirements specification, and the design criteria. Acceptance testing will include tests of all intrasystem interfaces; and the use of all manuals, documentation, procedures, and controls. This test phase can be performed by the project team with system owner and user observers or by system owner and user representatives with support from the project team.

Prototype Testing: In addition to the four test phases, a prototype or site test can be used when the system must be physically transported, installed, and made operational at a computer facility other than at the site(s) where the acceptance test was conducted. When required, this test is conducted at selected user location(s) that will totally test the product under "live" conditions with users and support personnel.

Note: Information about unit test, integration test, and system test may be documented within a single Project Test Plan.

Task: **4.6.3 Identify Test Environment Requirements**

Description: The Project Test Plan should outline what is needed to perform testing activities throughout the project lifecycle including personnel, hardware, software, space, and other environmental requirements. As much testing as possible should be performed on the same equipment that will be used for the production system. In many cases, this information is not fully known until the System Design Stage.

The following are some of the considerations for test environment requirements.

- Evaluate automated testing tools for the following:
  - Generation of test scripts
  - Creation of result and error repositories
  - Consideration of each tool's benefits and costs
  - Use of simulators
- Determine local area network, wide area network, and metropolitan area network testing environment(s), as needed
- Determine test lab, data generation, and error correction support
- Identify Beta test sites

Activity: **4.7 Develop Acceptance Test Plan**

Responsibility: Project Team

Description: The Acceptance Test Plan is a description of the test activities planned for project acceptance. The Acceptance Test Plan should establish the testing necessary to validate that the project requirements have been met and that the deliverables are at an acceptable level in accordance with existing standards. The plan also assures that a systematic approach to acceptance testing is established and that the testing is adequate to verify the functionality of the product.

The complete set of system requirements and acceptance criteria form the basis for determining the overall approach to acceptance testing and the specific testing and examination methods. Features of the installation site and the system affect how the acceptance testing will be done. Unique arrangements may be necessary when the product cannot be completely installed and executed in a live environment. Multiple configurations may have to be distributed at several installation sites.

When a new system is a replacement for one already in use, the acceptance test must assure the integrity of the users business operations while placing the replacement into operation. For example, the old system and the new system are used in parallel until complete functionality has been verified. In some cases, the acceptance process may take several months to assure that a complete business or accounting cycle has occurred. This concern will influence the approach to acceptance testing.

Work Product: Acceptance testing must be documented carefully with traceability of test cases to the requirements and acceptance criteria established by the system owner. As a minimum, the acceptance test plan should address the following requirements.

- Identification of the personnel involved in the acceptance test process and their testing responsibilities. If individuals outside of the project team perform acceptance testing, include the responsibilities and relationships of external test groups.
- Traceability of test designs and cases to requirements.
- The objectives and constraints for each test.

Work Product, continued:

- Complete test cases and test procedures including inputs and expected outputs for each test case.
- Descriptions of error reporting, analysis, and resolution.
- Location(s) where testing will occur, the testing approach, type of facilities, and tester training.
- Acquisition of special purpose testing equipment, tools, and software.
- Resources and cost estimation to accomplish testing.

Place a copy of the draft Acceptance Test Plan in the Project File. The draft plan will be reviewed during the Integration and Testing Stage and delivered as a final document.

Review Process: Conduct structured walkthroughs to assure the draft Acceptance Test Plan adequately describes all testing activities, test schedules, test products, test responsibilities, the testing methodology, and the required resources.

Resources: A template of the Acceptance Test Plan is available on the Software Quality & Systems Engineering web site.

Activity: **4.8 Select Design Technique**

Responsibility: Project Team

Description: A systematic approach for building the functional and system designs for the product simplifies the process and results in a product that is testable, reliable, and maintainable. A complete design technique includes the following elements:

- A technique that is compatible with the requirements analysis technique and any automated tools used by the project team.
- Straightforward rules that relate information obtained during requirements analysis to a distinct system structure.
- Design standards that comply with the site's current information systems engineering practices, the system owner organization's standards, and the constraints imposed by the software and hardware tools used by the project team.
- A practical approach to design that is amenable to a wide variety of products.
- The development of small, intermediate design products that can be used to measure quality and progress.
- An evolution process from functional to system design.
- Well-defined measures to assess the quality of the design.
- Guidance on how to detect and correct design features that reduce maintainability and reusability.

The value of a design technique can be significantly enhanced by automated tools that directly support the technique. Automated tools provide assistance in generating, maintaining, and analyzing design diagrams and data dictionaries. The use of such tools typically results in a design that is easier to maintain, higher in quality, and more complete than designs produced without automated tools. The increased quality leads to significant productivity gains during construction and testing.

Sample Design Methods: The following are examples of common design techniques.

- Function-oriented design methods model the product by breaking it into components, identifying the inputs required by those components, and identifying the outputs produced by them. Function-oriented design methods include structured analysis and structured design. The major models or design representations used by this method are data flow diagrams, data dictionaries, structure charts, and process specifications.

- Data-oriented design methods use program structures that are derived from the data structures. Tree diagrams are typically used to represent both the data and the program structures.

- Object-oriented design methods produce an architecture based on the objects manipulated by systems or subsystems rather than by functions. An object-oriented design closely resembles a model of reality since it captures the real-world objects and the operations taken by or upon them. The design structure tends to be layers of abstraction where each layer represents a collection of objects with limited visibility to other layers.

Work Product: Create a description of the design technique and distribute it to the project team, system owner, and users. Place a copy of the design technique description in the Project File.

Review Process: Conduct a structured walkthrough to verify that the design technique is appropriate for the scope and objectives of the project. A structured walkthrough is not needed when the technique has been used successfully on similar projects for the same system owner/user IT environment.

Chapter: **5.0 Functional Design Stage**

Description: The functional design process maps the "what to do" of the Requirements Specification into the "how to do it" of the design specifications. During this stage, the overall structure of the product is defined from a functional viewpoint. The functional design describes the logical system flow, data organization, system inputs and outputs, processing rules, and operational characteristics of the product from the user's point of view. The functional design is not concerned with the software or hardware that will support the operation of the product, or the physical organization of the data or the programs that will accept the input data, execute the processing rules, and produce the required output.

The focus is on the functions and structure of the components that comprise the product. The goal of this stage is to define and document the functions of the product to the extent necessary to obtain the system owner and users understanding and approval and to the level of detail necessary to build the system design.

Prototyping of system functions can be helpful in communicating the design specifications to the system owner and users. Prototypes can be used to simulate one function, a module, or the entire product. Prototyping is also useful in the transition from the functional design to the system design.

Input: The following work products provide input to this stage.

- Project File
- Configuration Management Plan *(draft)*
- Continuity of Operations Statement/Plan
- Data Dictionary (revised)
- Requirements Traceability Matrix
- Requirements Specification
- Project Test Plan
- Acceptance Test Plan *(revised)*
- Design methodology
- Project Plan *(revised)*
- Quality Assurance Plan

High-Level Activities: The remainder of this chapter is divided into sections that describe specific high-level activities performed during this stage. These activities represent the minimum requirements for a large information systems engineering effort. *Notes* are provided, as applicable, to assist in customizing these lifecycle stage requirements to accommodate different sizes of systems engineering efforts.

High-Level Activities, continued:

The high-level activities are presented in the sections listed below.

5.1 Determine System Structure
5.2 Design Content of System Inputs and Outputs
5.3 Design User Interface
5.4 Design System Interfaces
5.5 Design System Security Controls
5.6 Build Logical Model
5.7 Build Data Model
5.8 Develop Functional Design
5.9 Initiate Procurement of Hardware and Software

Output:

Several work products are developed during this stage. The work products listed below are the minimum requirements for a large systems project. Deviations in the context and delivery of these work products are determined by the size and complexity of a project. Explanations of the work products are provided under the applicable activities described in the remainder of this chapter.

- Design records
- Logical model
- Data Dictionary
- Requirements Traceability Matrix *(expanded)*
- Functional Design Document
- Minutes from Functional Design Review
- Hardware and software procurement records
- Project Plan *(revised)*

A matrix showing the work products associated with each high-level activity is provided in *Exhibit 5.0-1, Functional Design Stage Activities and Work Products by Project Size.* The matrix also shows which work products are deliverables and whether they are required or optional for small, medium, and large products.

Review the Project Plan for accuracy and completeness of all Functional Design Stage activities and make any changes needed to update the information.

Review Process:

Quality reviews are necessary during this stage to validate the product and associated work products. The activities that are appropriate for quality reviews are identified in this chapter and Chapter 2.0, Lifecycle Model. In addition, a Preliminary Design Review will be conducted. This review is an important milestone in the design process. The time and resources needed to conduct the walkthroughs and Functional Design Review should be reflected in the project resources, schedule, and work breakdown structure.

Review Process, continued:

Structured Walkthrough
Requirements for a peer review or a more formal structured walkthrough, are documented under *Review Process*, at the end of each Task, Subtask, or Activity section in this stage.

In-Stage Assessment
Schedule at least one ISA prior to the Functional Design Stage Exit process. Additional ISAs can be performed during the stage, as appropriate. The completion of the Functional Design Document is an appropriate time to schedule an ISA.

Stage Exit
Schedule a Stage Exit as the last activity of the Functional Design Stage to enable the project approvers to review project deliverables and provide a concur/non-concur position to the project manager.

References:

Chapter 2.0, Lifecycle Model, *Quality Reviews* provides an overview of the Quality Reviews to be conducted on a project.

Conducting Structured Walkthroughs provides a procedure and sample forms that can be used for structured walkthroughs.

In-Stage Assessment Process Guide provides a procedure and sample report form that can be used for in-stage assessments.

Stage Exit Process Guide provides a procedure and sample report form that can be used for stage exits.

Resource:

DOE Software Quality & Systems Engineering web site**.**

Bibliography:

The following materials were used in the preparation of the Functional Design Stage chapter.

1. Barker, Richard, *CASE*METHOD*, Tasks and Deliverables, 1990. pp. 5-10 and 5-11.

2. Gane, Chris and Sarson, Trish, *Structured Systems Analysis: Tools and Techniques*, Prentice-Hall, Inc., Englewood Cliffs, 1979.

3. *Software Engineering Handbook*, Chapter 4, Software Requirements Analysis. pp. 4-1 through 4-9.

Bibliography, Continued:

:

4. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Guide to Software Design Descriptions*, IEEE Std 1016.1-1993, New York, 1993.

5. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Guide to Software Requirements Specifications*, ANSI/IEEE Std 830-1998, New York, 1998.

6. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Lifecycle Processes*, IEEE Std 1074-1991, New York, 1992.

7. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Software Verification and Validation Plans*, ANSI/IEEE Std 1012-1998, New York, 1998.

8. U.S. Department of Defense, Military Standard, *Technical Reviews and Audits for Systems, Equipments, and Computer Software*, MIL-STD-1521 B, 1985. pp.5, 33-52.

9. U.S. Social Security Administration, Office of Systems, *Software Engineering Technology (SET) Manual*, Volume 1, 1990.

Exhibit 5.0-1. Functional Design Stage Activities and Work Products by Project Size

Work Activity		Project Size L	M	S	Work Product	Deliverables L	M	S
5.1	Determine System/Product Structure	R	R	R	Design Entities and Dependencies	I	I	I
5.2	Design Content of System Inputs and Outputs	R	R	R	System Input and Output Design	I	I	I
5.3	Design User Interface	R[2]	R[2]	R[2]	User Interface Design	I[2]	I[2]	I[2]
5.4	Design System Interfaces	R	R	R	System Interface Design	I	I	I
5.5	Design System Security Controls	R	R	R	System Security Control Design	I	I	I
5.6	Build Logical Model	R	R	R	Logical Model	R	R	R
5.7	Build Data Model	R	R	R	Data Dictionary *(revised)*	R	R	R
5.8	Develop Functional Design	R	R	R	Requirements Traceability Matrix *(expanded)* Functional Design Document Functional Design Review Minutes	R R N	R R N	R R N
5.9	Initiate Procurement of Hardware and Software	A	A	A	Procurement Records	N	N	N
3.8	Revise Project Plan	R	R	R	Project Plan *(revised)*	R	R	R
2.5	Conduct Structured Walkthrough(s)	R	R	A	Structured Walkthrough(s) Management Summary Report	N	N	N
2.5	Conduct In-Stage Assessment	R	R	A	ISA Report Form[1]	N	N	N
2.5	Conduct Functional Design Stage Exit	R	R	A	Stage Exit Meeting Summary	N	N	N

Size: L = Large, M = Medium, S = Small

Minimum Requirements: R = Required, A = As Appropriate, N = Not a Deliverable[3], I = Input to another Deliverable

O = Optional work product

[1] = Completed by reviewer

[2] = Can adapt an existing plan

[3]A deliverable is a work product that is identified in the Project Plan and is formally delivered to the system owner and other project stakeholders for review and approval.

Activity: **5.1 Determine System Structure**

Responsibility: Project Team Analysts

Description: A hierarchical approach is useful for determining the structure and components of the product. System decomposition is one hierarchical approach that divides the system into different levels of abstraction. Decomposition is an iterative process that continues until single purpose components (i.e., design entities or objects) can be identified. Decomposition is used to understand how the product will be structured, and the purpose and function of each entity or object.

The goal of the decomposition is to create a highly cohesive, loosely coupled, and readily adapted design. A design exhibits a high degree of cohesion if each design entity in the program unit is essential for that unit to achieve its purpose. A loosely coupled design is composed of program units that are independent or almost independent.

Several reliable methods exist for performing system decomposition. Select a method that enables the design of simple, independent entities. Functional and object-oriented design are two common approaches to decomposition. These approaches are not mutually exclusive. Each may be applicable at different times in the design process.

Tasks: The system decomposition activity includes the following tasks.

5.1.1 Identify Design Entities
5.1.2 Identify Design Dependencies

Task: **5.1.1 Identify Design Entities**

Description: Design entities result from a decomposition of the product requirements. A design entity is an element (or object) of a design that is structurally and functionally distinct from other elements and is separately named and referenced. The number and type of entities required to partition a design are dependent on a number of factors, such as the complexity of the product, the design method used, and the development environment. The objective of design entities is to divide the product into separate components that can be coded, implemented, changed, and tested with minimal effect on other entities.

Attributes: A design entity attribute is a characteristic or property of a design entity. It provides a statement of fact about an entity. The following are common attributes that should be considered for each design entity.

- Assign a unique name to each entity.
- Classify each entity into a specific type. The type may describe the nature of the entity, such as a subprogram or module; or a class of entities dealing with a particular type of information.
- Describe the purpose or rationale for each entity. Include the specific functional and performance requirements for which the entity was created.
- Describe the function to be performed by each entity. Include the transformation applied to inputs by the entity to produce the desired output.
- Identify all of the external resources that are needed by an entity to perform its function.
- Specify the processing rules each entity will follow to achieve its function. Include the algorithm used by the entity to perform a specific task and contingency actions in case expected processing events do not occur.
- Describe the data elements internal to each entity. Include information such as the method of representation, format, and the initial and acceptable values of internal data. This description may be provided in the data dictionary.

Work Product: Maintain a record of all design entities. The records will be integrated into the Functional Design Document. Place a copy of the design entity information in the Project File.

Review Process: Schedule structured walkthroughs to verify that the design entities are correct, complete, and possess the required attributes.

Task: **5.1.2 Identify Design Dependencies**

Description: Design dependencies describe the relationships or interactions between design entities at the module, process, and data levels. These interactions may involve the initiation, order of execution, data sharing, creation, duplication, use, storage, or destruction of entities.

Identify the dependent entities of the system design, describe their coupling, and identify the resources required for the entities to perform their function. Also define the strategies for interactions among design entities and provide the information needed to perceive how, why, where, and at what level actions occur.

Dependency descriptions should provide an overall picture of how the product will work. Data flow diagrams, structure charts, and transaction diagrams are useful for showing the relationship among design entities.

The dependency descriptions may be useful in producing the system integration plan by identifying the entities that are needed by other entities and that must be developed first. Dependency descriptions can also be used to aid in the production of integration test cases.

Work Product: Add specific dependency information to the design entity records. The records will be integrated into the Functional Design Document. Place a copy of the dependency information in the Project File.

Review Process: Schedule structured walkthroughs to verify that the design entities and dependencies are correct, complete, and possess the required attributes.

Activity: **5.2 Design Content of System Inputs and Outputs**

Responsibility: Project Team Analysts

Description: Design the content and format for each of the product inputs and outputs based on the system input and output requirements identified during the Requirements Definition Stage. Involve the system owner and users in the design process to make certain that their needs and expectations are being met.

Procedure: Use the following procedure to implement the design process.

- Identify the types of electronic and printed input that will be accepted by the product, such as data entered manually from source documents and files or records extracted from other systems.
- Identify the types of electronic and printed output that will be produced by the product; such as data, records, or files; screen displays; and printed reports. Also identify the output that will be exported to other systems.
- Identify the specific input and output items that already exist and the items that will be created for input or output as part of the product.
- Assign a name to each type of input and output and describe each item from a functional perspective.
- Identify the owner/originator of each type of input and output.
- Identify the frequency of each type of input and output.
- Design the content and format for each new input and output item or modify the format of existing items that must be changed to accommodate the new product.

Work Product: Document the design for the system inputs and outputs in accordance with the project design standards. Discuss the designs with the system owner and users and submit completed designs for their review and approval. The approved designs will be incorporated into the Functional Design Document. Place a copy of the system input and output designs in the Project File.

Review Process: Schedule a structured walkthrough to verify that the system input and output designs are correct and complete.

Activity: **5.3 Design User Interface**

Responsibility: Project Team Analysts

Description: Design a user interface that is appropriate for the users, content, and operating environment for the product. Determine interface levels for all categories of users. For interactive user environments, prototype the user interface. Arrange for users to experiment with the prototypes so that design weaknesses in the interface can be identified and resolved early. Use prototypes to gain user acceptance of the interface.

If the site or system owner's organization has an existing user interface standard, this standard should be used to specify the user interface for every product developed for that organization. A user interface standard should be developed and maintained for each organization that does not have one.

Review the standard each time a new product is planned to verify that the user interface is compatible with the product's selected system architecture. For example, some DOS-based user interface standards would not be appropriate for a Windows-based product.

Basic Principles: The following basic principles can help improve the product user interface when there is graphical, command-based, menu-driven, or block mode features.

- Give users control. Let them choose actions to perform.
- Give users feedback and progress reports. Tell them when the system is working and when an action is completed.
- Be consistent in the format and wording of text.
- Keep it simple. White space is as important on the screen as on the printed page. Reduce screen clutter.
- Use special effects carefully and sparingly. Be sure color screens also work in one color--some users are colorblind, and some users have monochrome monitors. Use color consistently. Beeps and other sounds can be annoying; so let users turn sound off.

Basic Principles, continued:

- Put information where it can be easily seen; avoid information in corners or borders.
- Limit the amount of information users must know. Offer choices instead of making users remember and manually enter information. Provide defaults, and make sure they are logical and satisfy a large number of users.
- Offer shortcuts. Keyboard shortcuts (e.g., hot keys) and command abbreviations help experienced users work more quickly.
- Help users get out of trouble. Provide messages that are understandable and that offer solutions.
- Let users reverse their actions. If an action will destroy something, identify the object of destruction and wait for a response.

Tasks: The following tasks are involved in specifying the user interface.

5.3.1 Design Menu Hierarchy
5.3.2 Design Data Entry Screens
5.3.3 Design Display Screens
5.3.4 Design Online Help
5.3.5 Design System Messages

Task: **5.3.1 Design Menu Hierarchy**

Description: Use the following guidelines to improve the design of menu hierarchies.

- Choose an organizing principle for the menu options, such as:
  - Expected frequency of use
  - Logical sequence of operations
  - Alphabetical order (should be used for horizontal word menus with five or more words)
- Put a meaningful title at the top of every menu.
- For full-screen menus, provide symmetric balance by centering the title and the menu options around the center axis of the screen.
- To facilitate scanning, put blank lines between logical groupings of menu options and after about every fifth option in a long list.
- Limit the number of menu choices to one screen.
- Select icons that are intuitive to the function they represent.
- Use a menu option selection method that is consistent with the technology available at the user's workstation and the size of the product being designed, such as:
  - Numbers
  - Letters or letter combinations
  - Cursor movement
- Provide a way for the user to leave the menu without performing any action. Be sure that the option to leave the menu describes the consequences of its selection.
- Words used for menu options should follow these rules:
  - Use words that clearly and specifically describe what the user is selecting.
  - Use common English words rather than computer or technical jargon. When space permits, spell out words completely.

Description, continued:

- Use simple, active verbs to tell users what actions will result from their choice. Try to start each option with a verb.
- Use parallel construction to describe the options.

- Minimize the highlighting used on a menu. Highlighting should be limited to situations where the user needs to know that there is an exception to the normal practice.
- Do not require the user to enter leading or trailing blanks or zeros, and do not include a default value on a menu.
- Display the menu options in mixed letters (i.e., upper and lower case).
- Organize menu hierarchies according to the tasks users will perform, rather than the structure of the modules.

Work Product: Document the design for the menu hierarchy in accordance with the project design standards. Discuss the design with the system owner and users and submit the completed design for their review and approval. The approved design will be incorporated into the Functional Design Document. Place a copy of the menu hierarchy design in the Project File.

Review Process: Conduct a structured walkthrough to ensure that the menu hierarchy design is complete and logical.

Task: **5.3.2 Design Data Entry Screens**

Description: Use the following guidelines to improve the design of data entry screens.

- When the user must transcribe data directly from a source document to the screen, the layout of the screen should be similar to the layout of the source document.
- Group data fields into logical categories on the screen; provide a header that describes the contents of each category.
- Make areas of the screen that are not needed for data entry or commands inaccessible to the user.
- Do not require the user to enter information that is already available to the software or can be computed by it.
- Do not require the user to enter dimensional units, leading or trailing blanks, or zeros.
- Allow the user to enter data by character replacement.
- Put a caption describing the data to be entered adjacent to each data field; incorporate memory joggers into the caption.
- Justify data entries automatically.
- Display default values in data fields when appropriate.
- Provide context-sensitive help for data entry fields.

Work Product: Document the designs for the data entry screens in accordance with the project design standards. Discuss the design with the system owner and users and submit the completed designs for their review and approval. The approved designs will be incorporated into the Functional Design Document. Place a copy of the data entry screen designs in the Project File.

Review Process: Conduct a structured walkthrough to assure that the data entry screen designs are consistent, complete, and logical.

Task: **5.3.3 Design Display Screens**

Description: Use the following guidelines to design display screens that are easy to use and understand.

- Put a title on every display screen. The title should clearly and specifically describe the contents of the screen.
- Display only information that the user needs to know.
- Display data to the user in directly usable form.
- Provide symmetric balance to displays by centering titles and headings and by placing information on both sides of the center axis.
- Every display should indicate how to exit from the screen. Use consistent exit procedures.
- When the display continues over multiple screens, the screen should indicate where the user is in the display (e.g., Screen 1 of 3).
- Data fields need to be grouped into logical categories or according to the structure of a source document (when there is one).
- Be consistent in the use of words and special characters.
- Display text conventionally in mixed letters (i.e., upper and lower case) and with appropriate punctuation. Avoid all uppercase letters. Put a blank line between paragraphs.
- Left justify text, and leave a ragged right margin.
- Avoid hyphenation of words between lines.
- Use abbreviations and acronyms only when they are significantly shorter than the full text and when they will be understood by the user.
- Be consistent with the format of information being displayed.
- Consider the skills of the users and the information they will manipulate when information is displayed in multiple windows.

Table and List Guidelines: Use the following guidelines to improve the design of online tables and lists.

- Put a meaningful label on the columns and, if appropriate, the rows of tables and lists. Continue the labels when a table or list extends over more than one screen.
- If data items are scrolled, the labels should be fixed on the screen and not be part of the scrolled area (they remain in place as the body of the table or list changes).
- If data items are continued on subsequent screens, the labels should be added to each screen.
- Arrange the items in a table or list in some recognizable order to facilitate scanning.
- Put items in a multiple column list in vertical columns that are read from left to right on the screen.
- Left justify columns of alphabetic data; right justify columns of numeric data or align them by the decimal point or other delimiter.
- Insert a blank line after about every fifth row in a long column.
- Insert a minimum of two spaces between the longest item in a column and the beginning of the next column.
- Start with a one (1) not a zero (0) when listed items are labeled by number.

Work Product: Document the design for the display screens in accordance with the project design standards. Discuss the designs with the system owner and users and submit the completed designs for their review and approval. The approved designs will be incorporated into the Functional Design Document. Place a copy of the display screen designs in the Project File.

Review Process: Conduct a structured walkthrough to ensure that the display screen designs are consistent, complete, and logical.

Task: **5.3.4 Design Online Help**

Description: Online help is typically requested by users when they want to perform a new, complex, or infrequently used procedure, or when they do not know what else to do. The text of online help messages needs to be planned, drafted, and evaluated as carefully as print documentation. In addition, the layout and format of online help must be designed to deal with the special constraints imposed by the video screen.

Use online help to explain concepts, procedures, messages, menu choices, commands, words, function keys, and formats. Work with the users to identify the level of detail needed for online help. Determine whether the users need a one-line message at the bottom of the screen or a full online explanation with successive levels of detail.

Effective online help messages tell users what the product is doing, where they are in the sequence of screens, what options they have selected, and what options are available.

Guidelines: The following guidelines can improve the design of online help.

- Write online help messages in plain English.
  - Straightforward and reads as if it were spoken.
  - Clear, direct, and simple.
  - Effectively organized with a concern for what users need to know.
- Address the user directly as "you"; use the active voice.
- Use simple action verbs to describe procedures. Do not use nouns to replace pronouns, verbs, and adjectives.
- Describe procedures in logical order.
- Avoid computer terms or other jargon, such as:
  - Terms that are unique to the computer profession or to a particular company.
  - Terms that have a common meaning outside of the data processing environment, but a special meaning within it, such as *boot*, *abort*, *default*, and *utility*.

Guidelines, continued:

- Terms that are created to describe some special function, such as *ungroup* and *dearchive*.

$ Avoid humor in online documentation.

- Write in short complete sentences and paragraphs and use proper punctuation.

- Write sentences in the positive or simple negative. Avoid the passive voice and do not use double negatives.

- Use bullets, numbered lists, and tables to make it easier to find the most important information. Leave ample open space.

  - Use bulleted lists to explain options. Whenever a sentence lists options with commas between them, consider breaking up the text into a bulleted list.

  - Use numbered lists to show the steps in a process.

  - Use a table to explain two or more categories of information.

$ Use examples to show users what they should enter and what the results will look like.

$ Do not expect users to read more than about three screens of help at one time.

$ Provide an orientation to the structure of the product.

$ Whenever possible display help text on the screen with the function or task that is being performed.

$ Provide a direct route back to the function or task being performed.

Work Product: Document the design for online help in accordance with the project design standards. Discuss the design with the system owner and users and submit the completed design for their review and approval. The approved design will be incorporated into the Functional Design Document. Place a copy of the online help design in the Project File.

Review Process: Conduct a structured walkthrough to ensure that the online help design is consistent, complete, and logical.

Task: **5.3.5 Design System Messages**

Description: System messages are the various types of information that the system provides to the user such as status messages, user prompts, and error messages.

Status Messages: Status messages are important for giving users the feeling they are in control of the system. They tell users what the system is doing, where they are in the sequence of screens, what options they have selected, and what options are available.

User Prompts: Prompts inform the user to type data or commands or to make a simple choice.

- Use prompts to ask the user to make a simple choice or to enter data or commands. Be as specific as possible.
- Include memory aids in the prompt to help users type a response in the proper format and order, initiate infrequently used processes, or clearly identify exceptions to normal practice.
- When defaults are allowed with prompts, indicate clearly which default value will be initiated.

Error Messages: Error messages should allow users to recover from mistakes by making it clear what the mistake was and how to correct it. Error messages need to be specific about why a mistake was made.

- Design the product to check for obvious errors.
- Be as specific as possible in describing the cause of an error. Do not use error codes.
- Do not assign blame to the user or the product in an error message. Use a neutral tone.
- Whenever possible, the error message should indicate what corrective action the user needs to take.
- Be consistent in the format, wording, and placement of messages.
- Consider describing error messages at more than one level of detail.

Work Product: Document the design for the system messages in accordance with the project design standards. Discuss the designs with the system owner and users and submit the completed designs for their review and approval. The approved designs will be incorporated into the Functional Design Document. Place a copy of the system message designs in the Project File.

Review Process: Conduct a structured walkthrough to ensure that the system message designs are consistent, complete, and logical.

Activity: **5.4 Design System Interfaces**

Responsibility: Project Team Analysts

Description: Develop a design depicting how the product will interface with other systems based on the system interface requirements identified in the Requirements Definition Stage. Submit the applicable interface designs for review by the system owner or system administrator for each system that will interface with the product. Any incompatibilities with the interfaces will be identified early in the design process and corrective actions can be initiated to assure each interface is properly designed and coded.

Sample Issues: The following list provides some of the issues that should be considered when designing the system interfaces.

- System inputs and outputs
- Method of interface
- Volume and frequency of data
- Platform of interfacing system
- Format of data
- Automatic or manual initiation of interface
- Need for polling device(s)
- Verification of data exchange
- Validation of data

Work Product: Document the design(s) for the system interfaces in accordance with the project design standards. Discuss the designs with the system owner and users and submit completed designs for their review and approval. The approved designs will be incorporated into the Functional Design Document. Place a copy of the system interface designs in the Project File.

Review Process: Schedule a structured walkthrough to verify that the system interface designs are correct and complete.

Activity: **5.5 Design System Security Controls**

Responsibility: Project Team Analysts and Security Personnel

Description: Design the security controls that will be incorporated into the product based on the security and access requirements identified during the Requirements Definition Stage. Design the security controls in conjunction with the site or system owner and the organization's Computer System Security Officer (CSSO).

Procedure: Use the following procedure to implement the design process.

- Identify the users and organizations that will have access to the product. Indicate what access restrictions they will have. All persons in a work area may not have the same security access level. Measures should be taken to assure that sensitive materials and systems requiring protection are not accessed by unauthorized individuals.

- Identify controls for the product, such as the user identification code for system access and the network access code for the network on which the product will reside.

- Identify whether access restrictions will be applied at the system, subsystem, transaction, record, or data element levels. Classified information must be protected in accordance with DOE directives.

- Identify physical safeguards required to protect hardware, software, or information from natural hazards and malicious acts.

- Identify communications security requirements.

Work Product: Document the design for the system security controls in accordance with the project design standards. Discuss the design with the system owner and users and submit the completed design for their review and approval. The approved design will be incorporated into the Functional Design Document. Place a copy of the system security control design in the Project File.

Review Process: Schedule a structured walkthrough to verify that the system security controls are correct and complete. Include the CSSO or the ACPPM in the walkthrough.

Resources:

- DOE Order 471.2A, Information Security Program, and DOE Manual 471.2-2, Classified Information Systems Security Manual.

- DOE Notice 205.1, Unclassified Cyber Security Program.

Activity: **5.6 Build Logical Model**

Responsibility: Project Team Analysts

Description: The logical model defines the flow of data through the system and determines a logically consistent structure for the system. Each module that defines a function is identified, interfaces between modules are established, and design constraints and limitations are described. The focus of the logical model is on the real-world problem or need to be solved by the product.

A logical model has the following characteristics:

- Describes the final sources and destinations of data and control flows crossing the system boundary rather than intermediate handlers of the flows.
- Describes the net transfer of data across the system boundary rather than the details of the data transfer.
- Provides for data stores only when required by an externally imposed time delay.

When building a logical model, the organization of the model should follow the natural organization of the product's subject matter. The names given to the components of the model should be specific. The connections among the components of the model should be as simple as possible.

Work Product: The logical model should be documented in user terminology and contain sufficient detail to obtain the system owner's and users' understanding and approval. Use data flow diagrams to show the levels of detail necessary to reach a clear, complete picture of the product processes, data flow, and data stores.

Maintain the logical model and data flow diagrams for incorporation into the Functional Design Document. Place a copy of the logical model and data flow diagrams in the Project File. Keep the logical model and diagrams up-to-date. They will serve as a resource for planning enhancements during the Maintenance Stage, particularly for enhancements involving new functions.

Review Process: Schedule a structured walkthrough to verify that the logical model is correct, logical, and complete.

Activity: **5.7 Build Data Model**

Responsibility: Project Team Analysts

Description: A data model is a representation of a collection of data objects and the relationships among these objects. The data model is used to provide the following functions:

- Transform the business entities into data entities.
- Transform the business rules into data relationships.
- Resolve the many-to many relationships as intersecting data entities.
- Determine a unique identifier (keys) for each data entity.
- Add the attributes (facts) for each data entity.
- Document the integrity rules required in the model.
- Determine the data accesses (navigation) of the model.

Work Product: The data dictionary started in the Requirements Definition Stage is expanded in this stage to catalog every known data element used in the user's work and every system-generated data element. Data elements are documented in detail to include attributes, known constraints, input sources, output destinations, and known formats.

The data dictionary can serve as a central repository of information for both developers and end users. The dictionary can include business rules, processing statistics, and cross-referencing information for multiple vendor environments.

To expand the data dictionary, define, analyze, and complete data definitions using the following steps.

- Identify data needs associated with various system features.
- Match (verify) data needs with the data dictionary.
- Match the data dictionary with specific data structures.

Work Product, continued:

- Create data record layouts.
- Ensure that all data can be maintained through add, change, or delete functions.

The data dictionary is further refined in the System Design Stage to complete the information on data elements, entities, files, physical characteristics, and data conversion requirements.

Sample Attributes: The following is a sample of the type of attributes (information) that should be included for each element in a data dictionary.

Long data name (full name)
Short data name (abbreviation)
Alias
Data definition
Owner(s)
Occurrence(s)/key
Program mode
Input source(s) (e.g., screens, external interfaces, system generated)
Output destination(s) (e.g., screens, reports, external interfaces)
Values/meanings
Protection/security
Default value
Length/precision
Character set (type)
Format
Range
Surface edits
Remarks

Review Process: Schedule a structured walkthrough to verify that the data dictionary is correct and complete. The data model for a software application should be validated against any Departmental or site specific data model.

Activity: **5.8 Develop Functional Design**

Responsibility: Project Team

Description: The functional design describes how the product will be structured to satisfy the requirements identified in the Requirements Specification. It is a description of the structure, components, interfaces, and data necessary before coding can begin.

The functional design is a model or representation of the product that is used primarily for communicating design information to facilitate analysis, planning, and coding decisions. It represents a partitioning of the system into design entities and describes the important properties and relationships among those entities. Design descriptions may be produced as documents, graphic representations, formal design languages, records in a database management system, and Computer Aided Systems Engineering (CASE) tool dictionaries.

Within the functional design, the design entities can be organized and presented in any number of ways. The goal of this activity is to compile the design entities and their associated attributes in a manner that facilitates the access of design information from various viewpoints (e.g., project management, configuration management, quality assurance, and testing). Also, the design entities and their attributes must be described in terms that are understandable to the system owner and users.

Work Product: Major work products are the Functional Design and the revised Requirements Traceability Matrix. Each requirement identified in the Requirements Specification must be traceable to one or more design entities. This traceability ensures that the product will satisfy all of the requirements and will not include inappropriate or extraneous functionality. Expand the Requirements Traceability Matrix developed in the Requirements Definition Stage to relate the functional design to the requirements.

Review Process: Conduct a structured walkthrough of the Functional Design and the Requirements Traceability Matrix.

Tasks: The following tasks are involved in developing the functional design.

5.8.1 Develop Functional Design Document
5.8.2 Conduct Functional Design Review

Task: **5.8.1 Develop Functional Design Document**

Description: The Functional Design Document defines the functions of the system in user terminology and provides a firm foundation for the development of the system design. The Functional Design Document should be written from the system owner/users' perspective. This document provides the owner/users with an opportunity to review and provide input to the product design before system design work is completed.

Work Product: Prepare a draft Functional Design Document. Use the designs developed for inputs, outputs, user and system interfaces, and security controls as input to this document. Submit the draft document to the system owner and users for their review and approval. After making the changes needed to resolve problems found during the review, the approved Functional Design Document becomes an official agreement and authorization to use the functional design as the basis for developing the system design. Place a copy of the approved Functional Design Document in the Project File.

Review Process: Conduct structured walkthroughs as needed to assure that the Functional Design Document is accurate, complete, and describes the functional design in a manner that can be understood by the system owner and users.

The completion of the draft Functional Design Document is an appropriate time to schedule an In-Stage Assessment (ISA).

Reference: The *In-Stage Assessment Process Guide* provides a description and instructions for conducting an ISA.

Resource: A template of the Functional Design Document is available on the Software Quality & Systems Engineering web site.

Task: **5.8.2 Conduct Functional Design Review**

Description: The Functional Design Review is a formal technical review of the basic design approach. The primary goal of the Functional Design Review is to demonstrate the ability of the information system design to satisfy the project requirements. The review should be a series of presentations by the project team to the system owner, users, functional area points-of-contact, and quality assurance representative. Vendors may be invited to participate in the Functional Design Review when an off-the-shelf software product or hardware item is being considered for the system architecture.

Conduct the Functional Design Review to perform the following verifications.

- Evaluate the progress, technical adequacy, and risk mitigation of the selected design approach. Determine whether the approved design approach is being followed by the project team.

- Evaluate the progress, technical adequacy, and risk mitigation of the selected test approach. Review the following items:

  - Project Test Plan
  - Organization and responsibilities of group conducting tests
  - Planned format, content, and distribution of test reports
  - Planned resolution of problems and errors identified during testing
  - Retest procedures
  - Change control and configuration management of test items
  - Special test tools not required as deliverables

- Evaluate the techniques to be used to meet quality assurance requirements.

- Establish the existence and compatibility of the physical and functional interfaces.

- Determine whether the functional design embodies all of the product requirements.

Description, continued:

- Verify that the design represents a system that can meet the functional, data, and interface requirements.
- Review the planned user interfaces to the system. Examples of the types of design information to review:
  - Operating modes for each display station. For each mode, the functions performed, and the displays and controls used.
  - The format and content standards for each screen (e.g., data locations, spaces, abbreviations, the number of digits, all special symbols, alert mechanisms).
  - Control and data entry devices and formats (e.g., keyboards, special function keys, and cursor control).
  - The format of all data inputs and provisions for error detection and correction.
  - The format for all status and error messages and data printouts (e.g., formats, headings, data units, abbreviations, spacing, columns).
- Demonstrate any rapid design prototypes used to make design decisions.
- Identify potential high risk areas in the design and any requirements changes that could reduce risk.
- Review to assure that consideration has been given to optimizing the maintainability and maintenance aspects of the product.

Review Items: The following items should be considered for review and evaluation during the Functional Design Review. Be prepared to discuss in technical detail any of these items within the scope of the review.

- Functional flows. Indicate how the system functional flows map the software and interface requirements to the individual high-level components of the product.

Review Items, continued:

- Storage allocation data. Describe the manner in which available storage is allocated to individual components. Timing, sequencing requirements, and relevant equipment constraints used in determining the allocation should be included.

- Control functions. Describe the executive control and start/recovery features of the product.

- Component structure. Describe the high-level structure of the product, the reasons for choosing the components, the development technique that will be used within the constraints of available computer resources, and any support programs that will be required in order to develop and maintain the product and allocated data storage.

- Security. Identify the security requirements and provide a description of the techniques to be used for implementing and maintaining security within the product.

- Information systems engineering facilities. Describe the availability, adequacy, and planned utilization of the information systems engineering facilities including both Government-provided and commercially available facilities.

- Information systems engineering facility versus the operational system. Describe any unique design features that exist in the functional design in order to allow use within the information systems engineering facility that will not exist in the operational product. Provide information on the design of support programs not explicitly required for the operational system that will be generated to assist in the development of the product.

- Development tools. Describe any special tools (e.g., simulation, data reduction, or utility tools) that are not deliverables, but are planned for use during systems development.

- Test tools. Describe any special test systems, test data, data reduction tools, test computer software, or calibration and diagnostic software that are not deliverables, but are planned for use during development.

Review Items, continued:

- Commercial resources. Describe commercially available computer resources, including any optional capabilities (e.g., special features, interface units, special instructions, controls, formats). Identify any limitations of commercially available equipment (e.g., failure to meet user interface, safety, and maintainability requirements) and identify any deficiencies.

- Existing documentation. Maintain a file and have available for review any existing documentation supporting the use of commercially available computer resources.

- Support resources. Describe the resources necessary to support the product during engineering, installation, and operational state (e.g., operational and support hardware and software personnel, special skills, human factors, configuration management, testing support, documentation, and facilities/space management).

- Standards. Describe any standards or guidelines that must be followed.

- Operation and support documentation. Describe the documentation that will be produced to support the operation and maintenance of the product.

Work Product: Create and distribute official meeting minutes for each session. The minutes should consist of significant questions and answers, action items and individual/group responsible, deviations, conclusions, and recommended courses of action resulting from presentations or discussions. Recommendations that are not accepted should be recorded along with the reason for non-acceptance. Minutes must be distributed to the system owner and users for review and notification of review performance as follows:

- Approval - indicates that the functional design is satisfactorily completed.

- Contingent Approval - indicates that the functional design is not considered accomplished until the satisfactory completion of identified action items.

- Disapproval - indicates that the functional design is inadequate. Another Functional Design Review is required, once specified changes to the functional design are completed.

Activity: **5.9 Initiate Procurement of Hardware and Software**

Responsibility: Project Manager/Team

Description: Careful consideration should be given to purchasing off-the-shelf products before expending the time, resources, and costs associated with developing custom-built systems. Whenever possible, acquire off-the-shelf products to satisfy some or all of the project requirements. In addition, some projects may require the acquisition of hardware or software to support the design, code, and test processes.

Try to acquire a demonstration package of any proprietary product before completing the design specifications. The proprietary product may prove inadequate or inappropriate once it has been evaluated through hands-on use. Create a pilot of the product to exercise the most important functions provided by the proprietary product as well as to obtain definite performance indications.

Initiate the procurement of any hardware or software well in advance of the planned need for these products. Adequate time must be allocated in the Project Plan timeline for the selection, procurement, installation, testing, and training associated with each vendor product.

The project team may assume all of the procurement, installation, and testing responsibilities, or the acquisition and testing of some hardware and software may be initiated by the functional area that is most familiar with the product. For example, a local area network engineering group may procure and test local area network or client/server software; a mainframe systems group may procure and test mainframe software.

Note: When the expected operating platform for a product will require extensive procurement of hardware and software, it is recommended that procurement needs be addressed as early in the lifecycle as possible.

Work Product: Place a copy of all software and hardware procurement records (e.g., justifications, approvals, purchase orders, and invoices) and the Acquisition and Installation Plans (if developed) in the Project File. If hardware and software acquisition requirements are known, develop the Acquisition and Installation Plans for all operating sites and initiate the procurement process. Review and, if necessary, revise the Acquisition and Installation Plans at the beginning of the Construction Stage. Requirements for the Acquisition and Installation Plans are provided in *Chapter 7, Construction Stage.*

Review Process: Not required; however, a peer review of software and hardware procurement records can be beneficial to ensure the correct order is placed. If the Acquisition plan is developed, then a review of the plan document should be conducted.

Chapter: **6.0 System Design Stage**

Description: The goal of this stage is to translate the user-oriented functional design specifications into a set of technical, computer-oriented system design specifications; and to design the data structure and processes to the level of detail necessary to plan and execute the Construction and Installation Stages. General module specifications should be produced to define what each module is to do, but not how the module is to be coded. Effort focuses on specifying individual routines and data structures while holding constant the structure and interfaces developed in the previous stage. Each module and data structure is considered individually during detailed design with emphasis placed on the description of internal and procedural details. The primary work product of this stage is a system design that provides a blueprint for the coding of individual modules and programs.

Input: The following items provide input to this stage.

- Project File
- Design records
- Logical model
- Data dictionary *(expanded)*
- Requirements Traceability Matrix *(expanded)*
- Functional Design Document
- Hardware and software procurement records
- Project Plan *(revised)*
- Quality Assurance Plan

High-Level Activities: The remainder of this chapter is divided into sections that describe the specific high-level activities performed during this stage. These activities represent the minimum requirements for a large information systems engineering effort. *Notes* are provided, as applicable, to assist in customizing these lifecycle stage requirements to accommodate different sizes of information systems engineering efforts. The high-level activities are presented in the sections listed below.

6.1 Select System Architecture
6.2 Design Specifications for Modules
6.3 Design Physical Model and Database Structure
6.4 Develop Integration Test Plan
6.5 Develop System Test Plan
6.6 Develop Conversion Plan
6.7 Develop System Design
6.8 Develop Program Specifications
6.9 Define Coding Practices

Output: Several work products are produced during this stage. The work products listed below are the minimum requirements for a large project. Deviations in the content and delivery of these work products are determined by the size and complexity of the project. Explanations of the work products are provided under the applicable activities described in the remainder of this chapter.

- Design specifications
- Physical Model
- Data Dictionary *(expanded)*
- Integration Test Plan *(draft)*
- System Test Plan *(draft)*
- Conversion Plan
- Requirements Traceability Matrix *(expanded)*
- System Design Document
- Program Specifications
- Coding Practices
- Project Plan *(revised)*

A matrix showing the work products associated with each high-level activity is provided in *Exhibit 6.0-1, System Design Stage Activities and Work Products by Project Size.* The matrix also shows which work products are deliverables and whether they are required or optional for small, medium, and large projects.

Review the Project Plan for accuracy and completeness of all System Design Stage activities and make any changes needed to update the information.

Review Process: Quality reviews are necessary during this stage to validate the product and associated work products. The activities that are appropriate for quality reviews are identified in this chapter and Chapter 2.0, Lifecycle Model. In addition, a Critical Design Review is conducted once the System Design Document is developed. The time and resources needed to conduct the walkthroughs and Critical Design Review should be indicated in the project resources, schedule, and work breakdown structure.

Structured Walkthrough
Requirements for a peer review or a more formal structured walkthrough are documented under *Review Process* at the end of each Task, Subtask, or Activity section in this stage.

In-Stage Assessment
Schedule at least one ISA prior to the System Design Stage Exit process. Additional ISAs can be performed during the stage, as appropriate. The completion of the System Design Document is an appropriate time to schedule an ISA.

Review Process, continued:

Stage Exit

Schedule a Stage Exit as the last activity of the System Design Stage to enable the project approvers to review project deliverables and provide a concur/non-concur position to the project manager.

References:

Chapter 2.0, Lifecycle Model, *Quality Reviews* provides an overview of the Quality Reviews to be conducted on a project.

Conducting Structured Walkthroughs, provides a procedure and sample forms that can be used for structured walkthroughs.

In-Stage Assessment Process Guide, provides a procedure and sample report form that can be used for in-stage assessments.

Stage Exit Process Guide, provides a procedure and sample report form that can be used for stage exits.

Resource:

DOE Software Quality & Systems Engineering web site.

Bibliography:

The following materials were used in the preparation of the System Design Stage chapter.

1. Barker, Richard, *CASE*METHOD,* Tasks and Deliverables, 1990. pp. 5-10 and 5-11.

2. Gane, Chris and Sarson, Trish, *Structured Systems Analysis: Tools and Techniques,* Prentice-Hall, Inc., Englewood Cliffs, 1979.

3. *Software Engineering Handbook* Chapter 5, Software Design; and Chapter 7, Software Testing.

4. *Software System Testing and Quality Assurance,* Chapter 5, Integration.

5. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Guide to Software Design Descriptions,* IEEE Std 1016.1-1993, New York, 1993.

6. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Guide to Software Requirements Specifications,* ANSI/IEEE Std 830-1998, New York, 1998.

7. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Recommended Practice for Software Design Descriptions,* IEEE Std 1016-1998, New York, 1998.

Bibliography, continued:

8. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Lifecycle Processes,* IEEE Std 1074-1991, New York, 1992.

9. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Software Verification and Validation Plans,* ANSI/IEEE Std 1012-1998, New York, 1998.

10. U.S. Department of Defense, Military Standard, *Defense System Software Development,* MIL-STD-2167A, 1988. pp. 27-29.

11. U.S. Department of Defense, Military Standard, *Specification Practices,* MIL-STD-490 B, 1986. pp. 47-50.

12. U.S. Department of Defense, Military Standard, *Technical Reviews and Audits for Systems, Equipments, and Computer Software,* MIL-STD-1521 B, 1985. pp.5, 33-52.

13. U.S. Department of Energy, *Automated Data Processing Systems Development Methodology, Volume 1,* K/CSD/INF/86-3, Vol.1, R3, prepared under contract by Martin Marietta Energy Systems, Inc. at Oak Ridge National Laboratory, August 1987.

14. U.S. Department of Energy, *Hanford Site Data Administration Guide,* RLO-ADP-1, July 1989.

15. U.S. Department of Energy, *Software Management Guide,* DOE/AD-0028, 1992.

16. U.S. Department of Energy, Nevada Operations Office, *Software Management Plan,* May 1991.

17. U.S. Social Security Administration, Office of Systems, *Software Engineering Technology (SET) Manual,* Volume 1, 1990. Part 40 Design Stage.

Exhibit 6.0-1. System Design Stage Activities and Work Products by Project Size

Work Activity		Project Size L	M	S	Work Product	Deliverables L	M	S
6.1	Select System Architecture	A	A	A	Analysis of Benefits and Costs Report Summary and Recommendations of Architecture Alternatives	A A	A A	A A
6.2	Design Specifications for Modules	R	R	R	Design Diagrams with Text	I	I	I
6.3	Design Physical Model and Database Structure	R	R	R	Data Dictionary *(expanded)* Physical Model	R R	R R	R R
6.4	Develop Integration Test Plan	R	R	R	Integration Test Plan *(draft)*	R	R	R
6.5	Develop System Test Plan	R	R	R	System Test Plan *(draft)*	R	R	R
6.6	Develop Conversion Plan	A	A	A	Conversion Plan	A	A	A
6.7	Develop System Design	R	R	R	Requirements Traceability Matrix *(expanded)* System Design Document Critical Design Review Minutes	R R N	R R N	R R N
6.8	Develop Program Specifications	R	R	R	Program Specifications	R	R	R
6.9	Define Coding Practices	R[2]	R[2]	R[2]	Coding Practices	R[2]	R[2]	R[2]
3.8	Revise Project Plan	R	R	R	Project Plan *(revised)*	R	R	R
2.5	Conduct Structured Walkthrough(s)	R	R	A	Structured Walkthrough Management Summary Report	N	N	N
2.5	Conduct In-Stage Assessment	R	R	A	ISA Report Form[1]	N	N	N
2.5	Conduct System Design Stage Exit	R	R	A	Stage Exit Meeting Summary	N	N	N

Size: L = Large, M = Medium, S = Small

Minimum Requirements: R = Required, A = As Appropriate, N = Not a Deliverable[3], I = Input to another deliverable

O = Optional work product

[1] = Completed by reviewer

[2] = Can adapt an existing plan

[3]A deliverable is a work product that is identified in the Project Plan and is formally delivered to the system owner and other project stakeholders for review and approval.

Activity: **6.1 Select System Architecture**

Responsibility: Project Team

Description: When the system architecture for the product has not been predetermined by the existing IT environment of the system owner and users, evaluate system architecture alternatives to determine which one has the best, cost-effective solution that satisfies the project requirements.

"Cost effective solution" does not imply the least expensive alternative. The best, cost effective solution is the alternative that does the best job of satisfying the project requirements, assures the highest quality product, and provides for an adequate return on investment in a timeframe that is acceptable to the system owner.

Select the specific hardware, software, database management system, and communication facilities based on the following types of considerations.

- DOE Enterprise Architecture or site-specific information architecture guidelines or standards
- Hardware and software that emphasizes simplicity, flexibility, ease of operation and maintenance
- Cost to procure and maintain potential environment
- Backup and recovery procedures
- Selection of a distributed or centralized processing environment
- Communication requirements
- Data configuration

Obtain support from functional area points-of-contact to aid in the architecture evaluation process. Consultations and input may be helpful from system and database administrators, local area network administrators, operations personnel, system developers, and communication experts.

Reference: The Departmental Information Architecture Program is described in Chapter 1.0, Introduction, in the *Relationship of the SEM to Other CIO and DOE Corporate Programs* section.

Resources:

- U.S. Department of Energy, *Information Architecture Volume I The Foundations*, DOE/HR-0141, 1995.
- U.S. Department of Energy, *Information Architecture II Baseline Analysis Summary*, DOE/HR-0171, 1996.
- U.S. Department of Energy, *Information Architecture Volume III Guidance*, DOE/HR-0178, 1997.
- U.S. Department of Energy, *Information Architecture Volume IV Vision*, DOE/HR-0190, 1998.
- U.S. Department of Energy, *Profile of Adopted Standards*, DOE/HR-0175, 1997.
- U.S. Department of Energy, *Information Architecture Standards Adoption and Retirement Process Service Action Plan*, DOE/HR-0173, 1997.
- U.S. Department of Energy, *Information Technology Standards Program Management Plan*, DOE/HR-0184, 1998.

The following documents provide detailed guidance on conducting an Analysis of Benefits and Costs and are available on the Software Quality & Systems Engineering web site.

- U.S. Department of Energy, *.Analysis of Benefits and Costs (ABC's) Guideline, Volume 1, A Manager's Guide to Analysis of Benefits and Costs.*
- U.S. Department of Energy, *Analysis of Benefits and Costs (ABC's) Guideline, Volume 2, An Analyst's Handbook for Analysis of Benefits and Costs.*

Tasks: The following tasks are involved in selecting a system architecture.

6.1.1 Evaluate System Architecture Alternatives
6.1.2 Recommend System Architecture

Task: **6.1.1 Evaluate System Architecture Alternatives**

Description: Consider system architecture alternatives within the site's information architecture guidelines that enable the project objectives and requirements to be achieved. The selection of a system architecture depends on many factors such as the experience of the project team with each alternative and the availability of reusable components to facilitate the implementation of an alternative.

When investigating alternatives, consider the following issues.

- Those functions or portions of functions that are to be automated and the functions that will be manual. Conduct an examination of *what* the automated portion of the project will encompass.
- The technical solution for the objectives. The determinations of *how* the product is to be designed; (e.g., online vs. batch, client-server vs. mainframe, Oracle vs. Sybase).
- The system owner's and users' IT environment and the needs created by the technical solution. Consider any hardware and software that must be acquired, including system access software, operating system software, database management system, and communications facilities.

The following procedure provides one approach for evaluating the architecture alternatives.

- Conduct an Analysis of Benefits and Costs to determine the most cost-effective alternative. On the benefits side, include the improvements over the current process being used to support the business application. On the costs side, include any degradation from current capabilities along with the rationale for allowing the degradation.
- Create and evaluate a data flow diagram for each alternative.
- Identify how users would interact with the features associated with each alternative (such as the generation of queries and reports).
- Create a list of the risks associated with each alternative and develop a plan for mitigating each risk.

Description, continued:

- Compare the performance capabilities of each alternative. How fast will each alternative be able to process the user's work given a particular hardware resource. Performance is usually expressed in terms of throughput, run time, or response time. Five factors that frequently affect performance include:

  - Number of intermediate files in a system (park data between programs)
  - Number of times a given file is passed
  - Number of seeks against a disk file
  - Time spent in calling programs and other system overhead
  - Time taken to execute actual program

- Compare the security and access control features of each alternative. To what extent does the alternative provide security against human errors, machine malfunction, or deliberate mischief. Some common controls include:

  - Check digits on predetermined numbers
  - Batch control totals
  - Creation of journals and audit trails
  - Limited access to files

- Compare the ease with which each alternative allows the system to be modified to meet changing requirements, such as:

  - Fixing errors
  - Changing user needs
  - Mandatory/statutory modifications
  - Enhancements

Work Product:

Maintain records on each alternative that is evaluated. Use this information to develop a summary of the system architecture alternatives. The summary will be integrated into the materials presented to the system owner when a system architecture recommendation is made. Place a copy of the records for each alternative and the summary in the Project File.

If an Analysis of Benefits and Costs (ABC) is conducted, prepare a report that describes the process used for the analysis, a summary of the alternatives considered, and the results obtained and place a copy in the Project File. The report will be integrated into the materials presented to the system owner when a system architecture recommendation is made.

Review Process: Conduct structured walkthroughs on records of each alternative that is evaluated.

Reference: For more information on conducting the Analysis of Benefits and Costs (ABC), refer to Chapter 3.0, *Planning Stage*.

Task: **6.1.2 Recommend System Architecture**

Description: Based on the results of the architecture alternatives evaluation, develop a recommendation for a system architecture that is cost-effective and will facilitate the achievement of the project requirements. Prepare a presentation for the system owner and users that provides the following types of information to support the recommendation.

- Review the limitations or problems with any current manual or IT system that will be resolved by the product.
- Present the logical model for the product. Highlight new functions that would be incorporated.
- For each architecture alternative that was evaluated, present the following information.
  - A description of the alternative.
  - An overall data flow diagram showing how the alternative would be implemented.
  - The way the system would look to the users, in terms of hardware, user interface, reports, and query facilities.
  - The estimated benefits of the alternative.
  - The estimated cost and time to implement the alternative.
  - A statement of the element of risk associated with the alternative.
- Present the recommended alternative and explain why it was selected.

Before the project proceeds, the system owner should make a decision about the system architecture either by formally accepting the project team's recommendation or by directing the team to use a different architecture. Any delay in making this decision could result in a slippage of the project schedule.

Work Product: Document the project team's recommendation for the most cost-effective and viable architecture alternative. Provide a summary of each alternative that was evaluated. Describe the rationale for proposing the recommended architecture. Describe the impact of this alternative on the system owner and users organization(s) and other systems. Include any background information that was relevant to the decision process, such as the Analysis of Benefits and Costs Report.

Review Process: Conduct a structured walkthrough to assure that the most cost-effective and viable architecture alternative is being recommended.

Approval: Present the project team's recommendation for the system architecture to the system owner and users. The recommendation can be delivered as a document or as a presentation. Place a copy of the document or presentation materials in the Project File.

Activity: **6.2 Design Specifications for Modules**

Responsibility: Project Team

Description: During the Functional Design Stage, a decomposition of the product requirements resulted in a collection of design entities (or objects). In the System Design Stage, these design entities are grouped into the routines, modules, and programs that need to be developed or acquired as off-the-shelf or reusable software.

Expand the functional design to account for each major action that must be performed and each data object to be managed. Detail the design to a level such that each program represents a function that a developer will be able to code.

Procedure: Use the following procedure to design the module specifications.

- Identify a program for each action needed to meet each function or data requirement in the Requirements Specification and the data dictionary.
- Identify any routines and programs that may be available as reusable code or objects from existing applications or off-the-shelf software.
- Identify programs that must be designed and developed (custom-built). Assign a name to each program and object that is functionally meaningful. Identify the system features that will be supported by each program.
- Specify each program interface. Update the data dictionary to reflect all program and object interfaces changed while evolving from the functional to the system design.
- Define and design significant attributes of the programs to be custom-built.
- Expand the program interfaces to include control items needed for design validity (e.g., error and status indicators).
- Combine similar programs and objects. Group the design entities into modules based on closely knit functional relationships. Formulate identification labels for these modules.

Procedure, continued:

- Show dependencies between programs and physical data structures (e.g., files and global tables). Avoid defining a program that not only needs data residing in a file or global table, but also depends on the physical structure or location of data.

- Change the design to eliminate features that reduce maintainability and reusability (i.e., minimize coupling between programs and maximize the cohesion of programs).

Work Product: Document the system design primarily in the form of diagrams. Supplement each diagram with text that summarizes the function (or data) and highlights important performance and design issues.

When using structured design methods, the design diagrams should:

- Depict the product as a top-down set of diagrams showing the control hierarchy of all programs to be implemented.

- Define the function of each program.

- Identify data and control interfaces between programs.

- Specify files, records, and global data accessed by each program.

When using object-oriented or data-centered design methods, the design diagrams should:

- Show the data objects to be managed by the product.

- Specify the program functions to be included within each object.

- Identify functional interfaces between objects.

- Specify files and records comprising each object.

- Identify relationships between data files.

Review Process: Conduct structured walkthroughs to assure that the custom-built routines and programs are correctly designed.

Activity: **6.3 Design Physical Model and Database Structure**

Responsibility: Project Team

Description: The physical model is a description of the dynamics, data transformation, and data storage requirements of the product. The physical model maps the logical model created during the Functional Design Stage to a specific technical reality. Care must be taken to retain in the physical implementation all of the capabilities inherent in the logical model.

The physical model frequently differs from the logical model in the following areas.

- Constraints imposed by the database management system - The logical data model may have different implementations in the selected database management system.
- Performance - Data redundancies, indices, and data structure changes may have to be introduced into the physical model to improve performance.
- Distributed processing - Possible network and multiple production hardware configurations may cause changes to the physical data model.

Designing the database structure converts the data requirements into a description of the master and transient files needed to implement the requirements. If the product will include a database, design the database in conjunction with the following database management features.

- Report writer and file processing capabilities
- Online query processing to retrieve data
- Automated data dictionary systems

Work Product: Document the physical model for incorporation into the System Design Document. Review the contents of the data dictionary entries and update to complete information on data elements, entities, files, physical characteristics, and data conversion requirements. Place a copy of all physical model and database structure records in the Project File.

Review Process: Schedule structured walkthroughs to verify that the physical model and data dictionary are correct and complete.

Activity: **6.4 Develop Integration Test Plan**

Responsibility: Project Team Developers

Description: The purpose of integration testing is to verify the integrity of a module (a cohesive set of programs) and its interfaces with other modules within the product structure. An integration test plan is developed to incorporate successfully unit-tested modules into the overall structure and to test each level of integration to isolate errors introduced by newly incorporated modules.

The number of integration levels, the classes of tests to be performed, and the order in which routines and builds are incorporated into the overall structure are addressed in the Integration Test Plan. The following factors should be considered.

- Are routines to be integrated in a pure top-down manner or should builds be developed to test subfunctions first?
- In what order should major functions be incorporated?
- Is the scheduling of module coding and testing consistent with the order of integration?
- Is special hardware required to test certain routines?

Integration testing should include tests that validate the following functions.

- Verify each interface between the module and all other modules.
- Access each input message or command processed by the module.
- Check each external file or data record referenced by coding statements in the module.
- Output each message, display, or record generated by the module.

An important consideration during integration test planning is the amount of test software (e.g., drivers, test case generation) that must be developed to adequately test the required functionality. For example, it may be cost-effective to delay testing of a communication function until hardware is available rather than generate test software to simulate communication links.

Description, continued: Similarly, it may be better to include certain completed modules in the structure in order to avoid having to develop software drivers. These decisions are made on the basis of cost and risks.

Work Product: Develop the draft Integration Test Plan that addresses the following activities.

- Define the integration tests at each element level, stating objectives, what is to be tested, and verified. Testing is from the point of view of structure and function.
- Define all aspects of the formal interfaces that must undergo formal integration testing. Review interface requirements to ensure completeness, consistency, and effectiveness.
- Plan for test tools and software that must be developed to adequately test the required functionality.

Note: The Integration Test Plan may be incorporated in the Project Test Plan.

Review Process: Conduct a peer review or structured walkthrough to assure that the draft Integration Test Plan is accurate and complete. The Integration Test Plan will be reviewed and revised as needed during the Construction Stage.

Activity: **6.5 Develop System Test Plan**

Responsibility: Project Test Team

Description: The objectives of the system test process are to assure that the product adequately satisfies the project requirements; functions in the computer operating environment; successfully interfaces between user procedures, operating procedures, and other systems; and protects the software and data from security risks. The system should be tested under the same kind of daily conditions that will be encountered during regular operations. System timing, memory, performance, and security functions are tested to verify that they perform as specified. The functional accuracy of logic and numerical calculations are tested for verification under normal and load conditions.

Test data should be varied and extensive enough to enable the verification of the operational requirements. Expected output results should be included in the test plan in the form of calculated results, screen formats, hardcopy output, predetermined procedural results, warnings, error messages and recovery.

Detailed planning for the system testing helps to ensure that system acceptance will be successfully completed on schedule. When applicable, system testing must include the following types of tests.

- Performance tests that measure throughput, accuracy, responsiveness, and utilization under normal conditions and at the specified maximum workload.
- Stress tests to determine the loads that result in appropriate, non-recoverable, or awkward system behavior.
- Interface tests to verify that the system generates external outputs and responds to external inputs as prescribed by approved interface control documentation.
- System recovery and reconfiguration tests.
- Verification that the system can be properly used and operated in accord with its users guide and operating instructions.
- Verification that the system meets its requirements for reliability, maintainability, and availability, including fault tolerance and error recovery.

Description, continued:

- Verification of the effectiveness of error detection and analysis, and automated diagnostic tools.
- Demonstration that the system complies with its serviceability requirements such as accessibility, logistics, upgrades, diagnostics, and repair capabilities.

Work Product: Develop a draft System Test Plan that describes the testing effort, provides the testing schedule, and defines the complete range of test cases that will be used to assure the reliability of the product. The test cases must be complete and the expected output known before testing is started. The test plan should address the following.

- Provide a definition of, and the objectives for, each test case.
- Define the test scenario(s) including the step-by-step procedure, the number of processing cycles to be tested or simulated, and the method and responsibility for feeding test data to the system.
- Define the test environment including the hardware and software environment under which the testing will be conducted. Identify and describe manual procedures, automated procedures, and test sites (real or simulated).
- Identify test tools and special test support needs (e.g., hardware and software to simulate operational conditions or test data that are recordings of live data).
- Identify responsibilities for conducting tests; for reviewing, reporting, and approving the results; and for correcting error conditions.
- Develop a requirements verification matrix mapping individual tests to specific requirements and specifying how each system requirement will be validated.
- Schedule for integrating and testing all components including adequate time for retesting.

Note: The System Test Plan may be incorporated into the Project Test Plan.

Review Process: Conduct peer reviews or structured walkthroughs to assure that each system test procedure is accurate, complete, and accomplishes the stated objectives. The System Test Plan will be reviewed and revised as needed during the Construction Stage.

Activity: **6.6 Develop Conversion Plan**

Responsibility: Project Team

Description: If the product will replace an existing IT system, develop a Conversion Plan. The major elements of the Conversion Plan are to develop conversion procedures, outline the installation of new and converted files/databases, coordinate the development of file-conversion, and plan the implementation of the conversion procedures.

File conversion should include a confirmation of file integrity. Determine what the output in the new system should be compared with the current system, and ensure that the files are synchronized. The objective of file conversion is new files that are complete, accurate and ready to use.

Many factors influence data conversion, such as the design of the current and new systems and the processes for data input, storage, and output. Understanding the data's function in the old system and determining if the function will be the same or different in the new system is of major importance to the Conversion Plan. The structure of the data to be converted can limit the development of the system and affect the choice of software.

Work Product: Develop a Conversion Plan that identifies what conversions are needed and how the conversion(s) will be implemented. Consider the following factors during the development of the conversion Plan.

- Determine if any portion of the conversion process should be performed manually.
- Determine whether parallel runs of the old and new systems will be necessary during the conversion process.
- Understanding the function of the data in the old system and determining if the use will be the same or different in the new system is important.
- The order that data is processed in the two systems influences the conversion process.
- Volume considerations, such as the size of the database and the amount of data to be converted, influence how the data will be converted. Especially important are the number of reads that are necessary, and the time these conversions will take.

Work Product, continued:

- User work and delivery schedules, timeframes for reports and end-of-year procedures, and the criticality of the data help determine when data conversion should be scheduled.
- Determine whether data availability and use should be limited during the conversion.
- Plan for the disposition of obsolete or unused data that is not converted.

Review Process: Conduct structured walkthroughs to assure that the Conversion Plan is accurate and complete.

Resource: A Conversion Plan template is available on the Software Quality & Systems Engineering web site.

Activity: **6.7 Develop System Design**

Responsibility: Project Team

Description: The system design is the main technical work product of the System Design Stage. The system design translates requirements into precise descriptions of the components, interfaces, and data necessary before coding and testing can begin. It is a blueprint for the Construction Stage, based on the structure and data model established in the Functional Design Stage.

The system design plays a pivotal role in the development and maintenance of a product. The design provides valuable information used by the project manager, quality assurance staff, configuration management staff, software designers, developers, testers, and maintenance personnel.

The system design is baselined after the system owner's formal approval of the design as described in the System Design Document. Once the system design is baselined, any changes to the design must be managed under change control procedures established in the Configuration Management Plan. Approved changes must be incorporated into the System Design Document.

It is important for the system owner/users to understand that some changes to the baselined system design may affect the project scope and therefore can change the project cost, resources, or schedule. It is the responsibility of the project manager and team to identify system owner/user requested changes that would result in a change of project scope; evaluate the potential impact to the project costs, resources, or schedule; and notify the system owner of the project planning revisions that will be required to accommodate their change requests.

Work Product: Major work products include the System Design and the updated Requirements Traceability Matrix. Each requirement identified in the Requirements Specification must be traceable to one or more design entities. This traceability ensures that the product will satisfy all of the requirements and will not include inappropriate or extraneous functionality. Expand the Requirements Traceability Matrix developed in the Requirements Definition Stage to relate the system design to the requirements.

Review Process: Conduct a structured walkthrough of the System Design and the Requirements Traceability Matrix.

Refer to task 6.7.2, *Conduct Critical Design Review,* for the system design review process.

Tasks: The following tasks are involved in developing the system design.

6.7.1 Develop System Design Document

6.7.2 Conduct Critical Design Review

Task: **6.7.1 Develop System Design Document**

Description: The System Design Document records the results of the system design process and describes how the product will be structured to satisfy the requirements identified in the Requirements Specification. The System Design Document is a translation of the requirements into a description of the structure, components, interfaces, and data necessary to support the construction process.

Work Product: Prepare the System Design Document and submit it to the system owner and users for their review and approval. The approved System Design Document is the official agreement and authorization to use the design to build the product. Approval implies that the design is understood, complete, accurate, and ready to be used as the basis for the subsequent lifecycle stages. In other words, once approved this becomes the design baseline. Subsequent changes or additions to the design that receive stakeholder concurrence supersede the existing baseline and establish a new design baseline. Place a copy of the approved System Design Document in the Project File.

Review Process: Conduct structured walkthroughs as needed to ensure that the System Design Document is accurate and complete.

The completion of the System Design Document is an appropriate time to schedule an In-Stage Assessment (ISA).

Reference: The *In-Stage Assessment Process Guide* provides a description and instructions for conducting an ISA.

Resource: A System Design Document template is available on the Software Quality & Systems Engineering web site.

Task: **6.7.2 Conduct Critical Design Review**

Description: The Critical Design Review is a formal technical review of the system design. The purpose of the review is to demonstrate to the system owner and users that the system design can be implemented on the selected platform and accounts for all software and data requirements and accommodates all design constraints (e.g., performance, interface, security, safety, resource, and reliability requirements). The design review should include a review of the validity of algorithms needed to perform critical functions.

Several short Critical Design Reviews can replace one long review if the product consists of several components that are not highly interdependent. The review process should be a series of presentations by the project team to the system owner and other approval authorities.

Conduct a Critical Design Review that demonstrates that the design specifications are capable of supporting the full functionality of the product, as follows:

- All algorithms will perform the required functions.
- The specification is complete, unambiguous and well documented, including timing and sizing, and data and storage allocations.
- The specification is necessary and sufficient for, and directly traceable to, the system design.
- The specification is compatible with every other specification, piece of equipment, facility, and item of system architecture, especially as regards information flow, control, and sequencing.
- The specification is consistent with the abilities of current development and user personnel.

In addition to verifying individual specifications, the Critical Design Review assesses other project work products to ensure the following.

- The approved design approach is being followed by the team.
- Measures to reduce risk on a technical, cost, and schedule basis are adequate.

Description, continued:

- The performance characteristics of the design solution are acceptable.
- Testing will be sufficient to ensure product correctness.
- The resultant application will be maintainable.
- Provisions for automatic, semi-automatic, and manual recovery from hardware/software failures and malfunctions are adequate and documented.
- Diagnostic programs, support equipment, and commercial manuals all comply with the system maintenance concept and specification requirements.

Work Product: Create and distribute official meeting minutes for each design review session. The minutes should consist of significant questions and answers, action items and individual/group responsible, deviations, conclusions, and recommended courses of action resulting from presentations or discussions. Recommendations that are not accepted should be recorded along with the reason for non-acceptance. Minutes must be distributed to review participants. The system owner determines review performance as follows:

- Approval - The review was satisfactorily completed.
- Contingent Approval - The review is not finished until the satisfactory completion of identified action items.
- Disapproval - The specification is inadequate. Another Critical Design Review will be required, once the required changes are made to the system design specifications.

Activity: **6.8 Develop Program Specifications**

Responsibility: Project Team

Description: A Program Specification is a written procedural description of each system routine. The Program Specification should provide precise information needed by the developers to develop the code.

Many techniques are available for specifying the system design, such as formal specification languages, program design languages (e.g, pseudo-code or structured English), meta-code, tabular tools (e.g., decision tables), and graphical methods (e.g., flow charts or box diagrams). In object-oriented design, the specification of requirements and preliminary design constraints and dependencies often results in the design language producing the detailed specifications.

Select the technique or combination of techniques that is best suited to the project and to the experience and needs of the developers who will use the system design as their blueprint. The following are suggestions for using the techniques.

- Decision trees are useful for logic verification or moderately complex decisions that result in up to 10-15 actions. Decision trees are also useful for presenting the logic of a decision table to users.

- Decision tables are best used for problems involving complex combinations of up to 5-6 conditions. Decision tables can handle any number of actions; however, large numbers of combinations of conditions can make decision tables unwieldy.

- Structured English is best used wherever the problem involves combining sequences of actions with decisions or loops. Once the main work of physical design has been done and physical files have been defined, it becomes extremely convenient to be able to specify physical program logic using the conventions of structured English, but without getting into the detailed syntax of any particular development language (pseudo-code).

- Standard English is best used for presenting moderately complex logic once the analyst is sure that no ambiguities can arise.

Work Product: Specifications may be produced as documents, graphic representations, formal design languages, records in a database management system, and CASE tool dictionaries. A list of program attributes typically included in a Program Specification is provided at the end of this section.

Review Process: Conduct a series of structured walkthroughs to ensure that the Program Specification is accurate and complete.

Sample Attributes: For each program to be custom built, define the program's functional and technical attributes as they become known. The following is a list of program attributes.

- Program identification
- Program name
- Program generic type
- Functional narrative
- Program hierarchical features diagram
- Development dependencies and schedule
- Operating environment
  - equipment
  - development language and version
  - preprocessor
  - operating system
  - storage restrictions
  - security
- Frequency of run
- Data volumes
- Program termination messages
  - normal termination
  - abnormal termination
- Console/printer messages
- Recovery/restart procedures
- Software objectives
- Program input/output diagram
- Data bank information
- Called and calling programs/modules
- Program logic diagrams
- Significant "how-to" instructions
- Telecommunications information

Activity: **6.9 Define Coding Practices**

Responsibility: Project Team Developers

Description: Coding practices are necessary to ensure that custom-built systems have acceptable design and structural properties. Coding practices must be practical, easy to implement, and accepted by the project team. The project team developers should be the primary producers of the standard. Use a structured approach to coding to allow for easy modification and to facilitate testing and debugging.

The following guidelines are generally applicable to any development language. Use these guidelines as the basis for the coding practices and add project-specific practices relating to the development language and tools.

- Control Flow Constructs
  - sequence
  - if-then-else
  - case statement
  - do-while (pretest loop)
  - do-until (post-test loop)
- Module Size
  - Number of executable lines of source code should average 100 lines per unit.
  - Units should contain no more than 200 lines of executable source code.
- Module Design
  - Units do not share temporary storage locations for variables
  - Units perform a single function
  - Avoid self-modifying code
  - Each unit is uniquely named
  - Each unit has a standard format:
    prologue
    variable declarations
    executable statements/comments
  - Use single entry/exit points except for error paths
  - Set comments off from the source code in a uniform manner

Description, continued:

- Symbolic Parameters
  - Use instead of specific numerics
  - Use for constants, size of data structures, relative position in list
- Naming Conventions
  - Use uniform naming throughout each unit and module to be put under configuration control
  - Use meaningful variable names
  - Do not use keywords as identifiers
- Mixed Mode Operations
  - Avoid mixed mode expressions
  - Add comments in code whenever used
- Error and Diagnostic Messages
  - Design messages to be self-explanatory and uniform
  - Do not require user to perform table lookups
- Style
  - Use conventions such as indentation, white space, and blank lines to enhance readability
  - Align compound statements
  - Avoid "goto" statements.
  - Avoid compound, negative Boolean expressions
  - Avoid nesting constructs beyond five levels deep
  - Avoid deeply nested "if" statements.
  - Use parentheses to avoid ambiguity
  - Include only one executable statement per line
  - Avoid slick development tricks that may create or encourage defects or be difficult to maintain; the most direct solution is best

Work Product: Create a coding practices document and distribute the document to all project team members. An existing set of practices can be used if it is applicable to the development language and tools being used for the project.

Review Process: Conduct a peer review to assure that the coding practices are complete and appropriate for the project's development language and tools.

Chapter: **7.0 Construction Stage**

Description: The goal of this stage is to translate the set of technical, computer-oriented system design specifications into a language the computer can understand and execute. Construction involves coding, validation and unit testing by a developer. Any hardware or software procured to support the construction effort is installed. Plans are developed for the acquisition and installation of the operating environment hardware and software. A training program is designed and a Training Plan that describes the system is produced.

The activities in this stage result in the transformation of the system design into the first complete executable representation of the product. If required, the source code or COTS "glue" code, including suitable comments, is generated using the approved program specifications. The source code is then grouped into processable units and all high-level language units are compiled into object code. Unit testing is performed to determine if the code satisfies the program specifications and is complete, logical, and error free.

The operating documentation is also produced. The operating documentation is required for installing, operating, and supporting the product through its lifecycle.

Input: The following items provide input to this stage.

- Project File
- Design specifications
- Physical model
- Data Dictionary
- Integration Test Plan *(draft)*
- System Test Plan *(draft)*
- Conversion Plan
- Requirements Traceability Matrix *(expanded)*
- System Design Document
- Program Specifications
- Coding Practices
- Project Plan *(revised)*
- Quality Assurance Plan

High-Level Activities: The remainder of this chapter is divided into sections that describe the specific high-level activities performed during this stage. These activities represent the minimum requirements for a large information systems engineering effort. *Notes* are provided, as applicable, to assist in customizing these lifecycle stage requirements

High-Level Activities, continued:

to accommodate the different sizes of information systems engineering efforts. The high-level activities are presented in the sections listed below.

7.1 Develop Acquisition Plan
7.2 Develop Installation Plan
7.3 Establish Construction Environment
7.4 Construct Programs
7.5 Conduct Unit Testing
7.6 Establish Development Baselines
7.7 Plan Transition to Operational Status
7.8 Generate Operating Documentation
7.9 Develop Training Program

Output:

Several work products are produced during this stage. The work products listed below are the minimum requirements for a large project. Deviations in the content and delivery of these work products are determined by the size and complexity of the project. Explanations of the work products are provided under the applicable activities described in the remainder of this chapter.

- Acquisition Plan
- Installation Plan *(draft)*
- System units and modules
- Requirements Traceability Matrix *(expanded)*
- Integration Test Plan *(final)*
- System Test Plan *(final)*
- Project Test File
- Development baselines
- Transition Plan
- Operating Documentation *(draft)*
  - Users Manual
  - Developer's Reference Manual
- Training Plan *(draft)*
- Project Plan *(revised)*

A matrix showing the work products associated with each high-level activity is provided in *Exhibit 7.0-1, Construction Stage Activities and Work Products by Project Size*. The matrix also shows which work products are deliverables and whether they are required or optional for small, medium, and large projects.

Review the Project Plan for accuracy and completeness of all Construction Stage activities and make any changes needed to update the information.

Review Process: Quality reviews are necessary during this stage to validate the product and associated work products. The activities that are appropriate for quality reviews are identified in this chapter and Chapter 2.0, Lifecycle Model. The time and resources needed to conduct the quality reviews should be reflected in the project resources, schedule, and work breakdown structure.

Structured Walkthrough
Requirements for a basic peer review or a more formal structured walkthrough are documented under *Review Process*, at the end of each Task, Subtask, or Activity section in this stage.

In-Stage Assessment
Schedule at least one ISA prior to the Construction Stage Exit process. Additional ISAs can be performed during the stage, as appropriate.

Stage Exit
Schedule a Stage Exit as the last activity of the Construction Stage to enable the project approvers to review project deliverables and provide a concur/non-concur position to the project manager.

References: Chapter 2.0, Lifecycle Model, *Quality Reviews* provides an overview of the Quality Reviews to be conducted on a project.

Conducting Structured Walkthroughs provides a procedure and sample forms that can be used for structured walkthroughs.

In-Stage Assessment Process Guide provides a procedure and sample report form that can be used for in-stage assessments.

Stage Exit Process Guide provides a procedure and sample report form that can be used for stage exits.

Resource: DOE Software Quality & Systems Engineering web site.

Bibliography: The following materials were used in the preparation of the Construction Stage chapter.

1. *Software Engineering Handbook*, Chapter 7, Software Testing.

2. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Lifecycle Processes,* IEEE Std 1074-1991, New York, 1992.

3. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Software User Documentation,* IEEE Std 1063-2001, New York, 2001.

Bibliography, Continued:

4. U.S. Department of Energy, *Automated Data Processing Systems Development Methodology, Volume 1,* K/CSD/INF/86-3, Vol.1, R3, prepared under contract by Martin Marietta Energy Systems, Inc. at Oak Ridge National Laboratory, August 1987.

5. U.S. Department of Energy, *DOE/NV Software Management Plan,* Nevada Operations Office, May 1991.

6. U.S. Department of Justice, Immigration and Naturalization Service, *System Development Lifecycle Standards Manual,* 1991.

7. The Institute of Electrical and Electronics Engineers, Inc., *Guide to the Software Engineering Body of Knowledge, Stone Man Trial Version 1.00,* New York, May 2001.

8. Carnegie Mellon University, Software Engineering Institute, *Capability Maturity Model: Guidelines for Improving the Software Process,* Addison Wesley Longman, Inc., 1994.

Exhibit 7.0-1. Construction Stage Activities and Work Products by Project Size

Work Activity		Project Size L	M	S	Work Product	Deliverables L	M	S
7.1	Develop Acquisition Plan	R	R	A	Acquisition Plan	R	R	A
7.2	Develop Installation Plan	R	R	A	Installation Plan *(draft)*	R	R	A
7.3	Establish Construction Environment	R	R	R				
7.4	Construct	R	R	R	Completed Units and Modules of Code Requirements Traceability Matrix (*expanded*)	I R	I R	I R
7.5	Conduct Unit Testing	R	R	R	Unit Test Materials Integration Test Plan *(final)* System Test Plan *(final)* Project Test File	I R R N	I R R N	I R R N
7.6	Establish Development Baselines	R	R	R	Baselined Code	R	R	R
7.7	Plan Transition to Operational Status	R	R	R	Transition Plan	R	R	R
7.8	Generate Operating Documentation	R	R	R	Users Manual *(draft)* Developer's Reference Manual *(draft)*	R R	R R	R R
7.9	Develop Training Program	R	R	A	Training Plan *(draft)*	R	R	A
3.8	Revise Project Plan	R	R	R	Project Plan *(revised)*	R	R	R
2.5	Conduct Structured Walkthrough(s)	R	R	A	Structured Walkthrough Management Summary Report	N	N	N
2.5	Conduct In-Stage Assessment	R	R	A	ISA Report Form[1]	N	N	N
2.5	Conduct Construction Stage Exit	R	R	A	Stage Exit Meeting Summary	N	N	N

Size: L = Large, M = Medium, S = Small

Minimum Requirements: R = Required, A = As Appropriate, N = Not a Deliverable[2], I = Input to other Deliverable

O = Optional work product

[1] = Completed by reviewer

[2]A deliverable is a work product that is identified in the Project Plan and is formally delivered to the system owner and other project stakeholders for review and approval.

Activity: **7.1 Develop Acquisition Plan**

Responsibility: Project Team

Description: Develop a plan for the acquisition of any hardware, software, and communications equipment needed to install and operate the product at all system owner and user sites. The plan should address any special procurements necessary to accommodate the hardware and communications equipment that may exist at a particular site. Acquisition planning must include sufficient lead time to accomplish all procurement, delivery, testing, and installation processes.

It may be necessary to perform a risk analysis of the impact of certain resources not being available when needed. Develop a contingency plan for dealing with needed resources that are acquired later than expected.

Work Product: Work closely with the system owner and representatives from the user sites to assure that all site-specific hardware, software, and communications needs are addressed in the Acquisition Plan.

Place a copy of the Acquisition Plan in the Project File.

Note: For projects that do not require extensive procurement and installation of hardware and software, the Acquisition and Installation Plans can be combined into one work product.

Review Process: Conduct a structured walkthrough to assure that the Acquisition Plan is accurate and complete.

Resource: An Acquisition Plan template is available on the Software Quality & Systems Engineering web site.

Activity: **7.2 Develop Installation Plan**

Responsibility: Project Team

Description: The Installation Plan is prepared to specify the requirements and procedures for the full-scale installation of the developed product at the system owner's and all users' work sites. The plan also addresses the installation of any hardware, off-the-shelf software, firmware, and communications equipment needed to operate the product at each site. In developing an Installation Plan consider each site's requirements for continuity of operations, level of service, and the needs of the project team, users, maintenance personnel, and management.

Work Product: Work closely with the system owner and representatives from the user sites to assure that all site-specific hardware, software, and communications installation requirements are addressed in the Installation Plan. Develop a draft Installation Plan that addresses the following issues.

- Schedule of all installation activities.
- Items to be delivered to each installation site.
- Number and qualifications of personnel performing installation.
- Equipment environmental needs and installation instructions.
- Hardware, software, firmware, tools, documentation, and space required for each installation.
- Special requirements governing the movement of equipment to each site.
- Communications requirements.
- Dependencies among activities affected by installation.
- Installation tests to assure the integrity and quality of the installed product.

Ensure any special requirements regarding, e.g., network resources and web-based applications are adequately documented. Place a copy of the draft Installation Plan in the Project File.

Note: For projects with limited procurement and installation requirements, the Acquisition and Installation Plans can be combined into one work product.

Review Process: Conduct a structured walkthrough to assure that the draft Installation Plan is accurate and complete. The Installation Plan will be reviewed and revised as needed during the Integration and Testing Stage.

Resource: An Installation Plan template is available on the Software Quality & Systems Engineering web site.

Activity: **7.3 Establish Construction Environment**

Responsibility: Project Team

Description: Establishing the construction environment involves assembling and installing the hardware, software, communications equipment, databases, and other items required to support the coding/construction effort. When the installation of the equipment or software is complete, conduct testing to verify the operating characteristics and functionality of the hardware and software. If required, security software and procedures should be activated when the installations are completed.

If the operational environment is also the construction environment, it may be necessary to alter the operational environment to accommodate an infrastructure of purchased hardware and software for use during construction and testing.

Before being integrated into, or used to support, the product, vendor products should be tested to verify that the product satisfies the following objectives.

- The product performs as advertised/specified.
- The product's performance is acceptable and predictable in the target environment (e.g., testing for LAN certification).
- The product fully or partially satisfies the project requirements.
- The product is compatible with the project team's other hardware and software tools.

Time should be planned for the project team to become familiar with new products. Ensure that the project team members who will use the hardware or software obtain proper training. This may involve attendance at formal training sessions conducted by the vendor or the services of a consultant to provide in-house training.

This is a good time to review the coding practices that were established in the System Design Stage. Make any changes to the standards that are needed to accommodate the procured hardware and software.

Activity: **7.4 Construct Programs**

Responsibility: Project Team Developers

Description: This activity involves generating the source and object code for the product. The code should be written in accordance with the coding practices developed in the System Design Stage. The use of various development languages will dictate whether the code is manually prepared or automatically generated. Regardless of the platform, construction of the code should adhere to a consistent set of coding practices and error prevention procedures. This will promote reliable, maintainable code, developed in the most efficient and cost effective manner.

Code units should be generated in a sequence based on a plan that accounts for factors such as criticality, difficulty, integration and test issues, and needs of customers and users, as appropriate.

For COTS products, although software COTS products attempt to simulate the "plug and play" capability of the hardware world, frequently this is not the case. Most products require some amount of adaptation and integration to work within the DOE environment. The typical solution is to adapt each software COTS product through the use of "wrappers," "bridges," or other "glueware." It is important to note that adaptation does not imply modification of the COTS product. Adaptation can be a complex activity that requires technical expertise at the detailed system and specific COTS component levels. Adaptation and integration must take into account the interactions among custom components, COTS products, any non-developmental item components, any legacy code, and the architecture including infrastructure and middleware elements.

The source and object code should be uniquely identified and stored in a way to facilitate the configuration control measures described in the Configuration Management Plan.

Constructing programs includes the following tasks.

- Use the Program Specifications developed in the System Design Stage as the basis for the coding effort.
- Generate source code.
- Generate the physical files and database structure.
- Generate video screens and report generation codes.
- Configure predefined options for COTS products.

Description, *continued:* If conversion of an existing system or data is necessary, generate the program(s) described in the Conversion Plan.

- Conduct preliminary testing of completed units. When the test output is correct, review the program specification to assure that the unit or module conforms to the specification.

Coding Practices: The following coding practices should be implemented.

- The code development staff should meet at scheduled intervals to discuss problems encountered and to facilitate program integration and uniformity.
- Program uniformity should be achieved by using a standardized set of naming conventions for programs, data elements, variables, and files.
- Modules that can be shared by programs requiring the same functionality should be implemented to facilitate development and maintenance.
- Modules that can be borrowed or reused from other sources that have already been tested should be implemented. Code reuse can lead to faster development.
- Meaningful internal documentation should be included in each program.
- All code should be backed up on a daily basis and stored in an offsite location to avoid catastrophic loss.
- A standard format for storing and reporting elements representing numeric data, dates, times, and information shared by programs should be determined.
- The System Design Document should be updated to reflect any required deviations from the documented design.

Work Products: The following work products are produced.

- Completed units and modules of code.
- Configured COTS options (e.g., databases, tables)
- Test materials generated from preliminary testing.

Work Products, continued:

Each requirement identified in the Requirements Specification must be traceable to the code. This traceability ensures that the product will satisfy all of the requirements and will not include inappropriate or extraneous functionality. Expand the Requirements Traceability Matrix developed in the Requirements Definition Stage to relate the source and object code to the requirements. Place a copy of the expanded matrix in the Project File.

Review Process: Periodic informal reviews of each developer's work are recommended to keep the project team informed of progress and to facilitate the resolution of any problems that may occur. The combined knowledge and skills of the team members will help to build quality into the product and support the early detection of errors in design, logic, or code.

Conduct structured walkthroughs on the expanded Requirements Traceability Matrix and completed units and modules to assure that the code is accurate, logical, internally well documented, complete, and error free. Structured walkthroughs should also be used to validate that the code is reliable and satisfies the program specifications and project requirements.

For large or complex projects, code inspections may be a more effective method of removing defects and identifying areas where defects may be propagated. Conduct the code inspections at successive stages of code production. Code inspection is a static analysis technique that relies on visual examination of code to detect errors, violations of development standards, and other problems. These inspections are particularly important when code is being produced by several developers or different teams. The inspection team may include experts outside of the project. Ideal times for code inspections occur when code and unit tests are complete, and when the first integration tests are complete. Code inspections should be identified as milestones in the Project Plan.

Activity: **7.5 Conduct Unit Testing**

Responsibility: Project Team Developers

Description: Unit testing is used to verify the input and output for each module. Successful testing indicates the validity of the function or sub-function performed by the module and shows traceability to the design. During unit testing, each module is tested individually and the module interface is verified for consistency with the design specification. All important processing paths through the module are tested for expected results. All error handling paths are also tested.

Unit testing is driven by test cases and test data that are designed to verify requirements, and to exercise all program functions, edits, in-bound and out-of-bound values, and error conditions identified in the program specifications. If timing is an important characteristic of the module, tests should be generated that measure time critical paths in average and worst-case situations.

Plan and document the inputs and expected outputs for all test cases in advance of the tests. Log all test results. Analyze and correct all errors and retest the unit using the scenarios defined in the test cases. Repeat testing until all errors have been corrected.

While unit testing is generally considered the responsibility of the developer, the project manager or lead developer should be aware of the unit test results.

Work Products: Completion of unit testing for a component signifies internal project delivery of a component or module for integration with other components. Place all components that have completed unit testing under configuration control as described in the Configuration Management Plan. These components form the Production Baseline. Configuration controls restrict changes to tested and approved software in the Production Baseline. Subsequent changes or additions to the software that are agreed upon in a Critical Design Review and receive stakeholder concurrence supersede the existing baseline and establish a new Production Baseline.

Review the draft versions of the Integration and System Test Plans developed during the System Design Stage. Update the plans, as needed, to reflect any changes made to the design. Deliver the final versions of the Integration and System Test Plans to the system owner and user for review and approval. Place a copy of the approved plans in the Project File.

Work Products, continued:

Create a Project Test File for all test materials generated throughout the project lifecycle. Place all unit test materials (e.g., inputs, outputs, results and error logs) in the Project Test File. The test cases used for unit testing may become a subset of tests for integration testing.

Review Process:

Conduct peer reviews on the test materials to be placed in the Project Test File. Conduct structured walkthroughs on any updated plans, (e.g., Integration and System Test Plans.)

Activity: **7.6 Establish Development Baselines**

Responsibility: Project Team Developers

Description: A development baseline is an approved "build" of the product. A build can be a single component or a combination of components. The first development baseline is established after the first build is completed, tested, and approved by the project manager or lead developer. Subsequent versions of a development baseline should also be approved. The approved development baseline for one build supersedes that for its predecessor build.

Conduct internal build tests such as regression, functional, performance, and reliability. Regression tests are designed to verify that capabilities in earlier builds continue to work correctly in subsequent builds. Functional tests focus on verifying that the build meets its functional and data requirements and correctly generates each expected display and report. Performance and reliability tests are used to identify the performance and reliability thresholds of each build.

Once the first development baseline is established, any changes to the baseline must be managed under the change control procedures described in the Configuration Management Plan. Approved changes to a development baseline must be incorporated into the next build of the product and revisions made to the affected work products (e.g., Requirements Specification, System Design Document, and Program Specifications).

Work Product: Document the internal build test procedures and results. Identify errors and describe the corrective action that was taken. Place a copy of the internal build test materials in the Project Test File.

Maintain configuration control logs and records as required in the Configuration Management Plan.

Expand the Requirements Traceability Matrix developed in the Requirements Definition Stage. All work products developed during the code, unit testing, and build processes must be traced back to the project requirements and system design. This traceability ensures that the product will satisfy all of the requirements and remain within the project scope. Place a copy of the expanded Requirements Traceability Matrix in the Project File.

Review Process: Conduct peer reviews on the internal build test materials to be placed in the Project Test File. Conduct structured walkthroughs on any updated documents, (e.g., the Requirements Traceability Matrix.)

Activity: **7.7 Plan Transition to Operational Status**

Responsibility: Project Team

Description: Successful transition from acceptance testing to full operational use of the product depends on planning the transition long before the product is installed in its operational environment. In planning for the transition, quantify the operational needs associated with the product and describe the procedures that will be used to perform the transition. Rely on experience and data gathered from previous, similar projects to define these needs.

Work Product: Develop a Transition Plan that describes the detailed plans, procedures, and schedules that will guide the transition process. Coordinate development of the plan with the operational and maintenance personnel. The following issues should be considered in the preparation of a Transition Plan.

- Develop detailed operational scenarios to describe the functions to be performed by the operational support staff, maintenance staff, and users.
- Define the number and qualifications of operations and maintenance personnel and specify when they must be in place. Estimate training requirements for these people.
- Document the release process. If development is incremental, define the particular process, schedule, and acceptance criteria for each release.
- Describe the development or migration of data, including the transfer or reconstruction of historic data. Schedule ample time for the system owner and user to review the content of reconstructed or migrated data files to reduce the chance of errors or omissions.
- Specify problem identification and resolution procedures for the operational product.
- Define the configuration management procedures that will be used for the operational product. Ideally, the methods defined in the Configuration Management Plan that were employed during product development can continue to be used for the operational product.

Work Product, continued:

- Define the scope and nature of support that will be provided by the project team during the transition period.
- Specify the organizations and individuals who will be responsible for each transition activity, ensuring that responsibility for the product by the operations and maintenance personnel increases progressively.
- Identify products and support services that will be needed for day-to-day operations or that will enhance operational effectiveness.

Review Process: Conduct a structured walkthrough to assure that the Transition Plan is logical, accurate, and complete. Involve operational and maintenance personnel in the walkthrough.

Resource: A Transition Plan template is available on the Software Quality & Systems Engineering web site.

Activity: **7.8 Generate Operating Documentation**

Responsibility: Project Team/Technical Writer

Description: Plan, organize, and write the operating documentation that describes the functions and features of the product from the users point-of-view. The different ways that users (including system administration and maintenance personnel) will interact with the product must be considered. The needs of the users should dictate the document presentation style and level of detail. Responsibilities for changing and maintaining the documents should be described in each document.

The following are typical operating documents for a large project.

- Users Manual/Online Help Screens
- Developer's Reference Manual
- Systems Administration Manual
- Database Administration Manual
- Operations Manual

It is recommended that a technical writer be involved in the generation of all operating documents. A technical writer works closely with the project team to ensure that documents are grammatically correct; comply with applicable standards; and are consistent, readable, and logical.

Note: The operating documents can be produced as separate manuals or combined to accommodate less complex projects.

Procedure: Use the following procedure to develop the operating documentation.

- Identify the operating documents that need to be developed. Determine if any of the documents can be combined or delivered as multiple volumes.

- Determine whether the documents should be provided as printed material, standalone electronic files, online documentation accessed through the product, or a combination.

- Determine the best presentation method or combination of methods required for each of the documents, such as a traditional manual, quick reference guide or card, or online help.

Procedure, continued:

- Identify all of the features of the user interface and the tasks users will perform.
- Identify the users' needs and experience levels to determine:
  - The amount of user interaction, level of interaction, and whether the interaction is direct or indirect.
  - The appropriate level of detail (e.g., the Users Manual should not contain highly technical terms and explanations that may confuse or frustrate a user).
- Determine the document content and organization based on whether the document will be used more as an instructional tool or a reference guide.
- Develop descriptions of each function and feature of the product and organize the information to facilitate quick, random access.
- Provide appropriate illustrations and examples to enhance clarity and understanding.
- Establish a schedule for the documents to be reviewed after the product goes into production. Operating documents must be kept up-to-date as long as the product remains in production.

Work Products: Refer to each of the following tasks for applicable work products.

Review Process: Refer to each of the following tasks for applicable review processes.

Tasks: The following tasks describe the minimum requirements for operating documentation.

7.8.1 Produce Users Manual
7.8.2 Produce Developer's Reference Manual

Task: **7.8.1 Produce Users Manual**

Description: The Users Manual provides detailed information users need to access, navigate through, and operate the product. Users rely on the Users Manual to learn about the product or to refresh their memory about specific functions. A Users Manual that is organized functionally so that the information is presented the same way the product works helps users understand the flow of menus and options to reach the desired functions.

Different categories of users may require different types of information. A modular approach to developing the Users Manual to accommodate the needs of different types of users eliminates duplication and minimizes the potential for error or omission during an amendment or update. For example, separate general information that applies to all users from the special information that applies to selected users such as system administrators or database administrators. The special information can be presented in appendixes or supplements that are only provided to the users who need the information.

Work Product: Write the draft Users Manual in clear, non-technical terminology that is oriented to the experience levels and needs of the user(s). The following are typical features of a users manual.

- Overview information on the history and background of the project and the architecture, operating environment, and current version or release of the product.
- Instructions for how to install, setup, or access the product.
- Complete coverage of all functions, presented in a logical, hierarchical order.
- Accurate pictures of screens and reports, ideally with data values shown, so the user can easily relate to examples.
- In-depth examples and explanations of the areas of the product that are most difficult to understand.
- Clear delineation of which features are accessible only to specific users.
- Instructions on accessing and using online help features.
- Procedures for data entry.

Work Product, continued:

- Descriptions of error conditions, explanations of error messages, and instructions for correcting problems and returning to the function being performed when the error occurred.
- Instructions for performing queries and generating reports.
- Who to contact for help or further information.

Note: For large or complex products, separate manuals (e.g., User's Manual, Database Administrator's Manual, and System Administrator's Manual) may be necessary to address the needs of different categories of users.

For very small projects, a quick reference guide or card may be more appropriate than a full-scale Users Manual. The guide or card should be designed to provide a quick reference of logon, logoff, and commands for frequently used functions.

For projects of any size, a quick reference card may be developed as a supplement to more detailed user documentation.

Review Process: Conduct structured walkthroughs for the draft Users Manual or set of user documents to assure that the documentation is complete, easy to use, and accurately reflects the product and its functions.

The draft user documentation will be tested and verified with the product during the Integration and Testing Stage.

Task: **7.8.2 Produce Developer's Reference Manual**

Description: The Developer's Reference Manual contains information about program development used by the maintenance staff to maintain the programs, databases, interfaces, and operating environment. The Developer's Reference Manual should provide an overall conceptual understanding of how the product is constructed and the details necessary to implement corrections, changes, or enhancements.

The Developer's Reference Manual describes the logic used in developing the product and the functional and system flow to help the maintenance staff understand how the programs fit together. The information should enable a developer to determine which programs may need to be modified to change a system function or to fix an error.

Work Product: The following are typical features of a Developer's Reference Manual.

- A description of the technical environment, including versions of the development language(s) and other proprietary software packages.
- A brief description of the design features including descriptions of unusual conditions and constraints.
- An overview of the architecture, program structure, and program calling hierarchy.
- The design and coding practices and techniques used to develop the product.
- Concise descriptions of the purpose and approach used for each program.
- Layouts for all data structures and files used in the product.
- Descriptions of maintenance procedures, including configuration management, program checkout, and system build routines.
- The instructions necessary to compile, link, edit, and execute all programs.
- Manual and automated backup procedures.
- Error-processing features.

Work Product, continued:

Use appendixes to provide detailed information that is likely to change as the product is maintained. For example, a list of program names and a synopsis of each program could be included as an appendix.

Review Process:

Conduct structured walkthroughs of the draft Developer's Reference Manual to assure that the documentation is complete, easy to use, and accurately reflects the product and its functions.

The draft Developer's Reference Manual will be tested and verified with the product during the Integration and Testing Stage.

Activity: **7.9 Develop Training Program**

Responsibility: Project Team

Description: A Training Program defines the training needed to implement and operate the product successfully. The Training Plan should address the training that will be provided to the system owner, users, and maintenance staff. When new hardware or software is being used, affected personnel will need hands-on experience before bringing the new system (equipment and/or software) into daily operation.

Training must address both the knowledge and the skills required to operate and use the system effectively. Design the training program to accomplish the following objectives.

- Provide trainees with the specific knowledge and skills necessary to perform their work.
- Prepare training materials that will sell the product as well as instruct the trainees. The training program should leave the trainees with the enthusiasm and desire to use the new product.
- Account for the knowledge and skills the trainees bring with them, and use this information as a transition to learning new material.
- Anticipate the needs for follow-on training after the product is fully operational, including refresher courses, advanced training, and repeats of basic courses for new personnel.
- Build in the capability to update the training as the product evolves.

Involve the system owner and key users in the planning to determine the education and training needs for all categories of users (managers, users, and maintenance staff).

The Training Program should address the following issues:

- Identify the organization's training policy for meeting training needs.
- Ensure software managers have received orientation on the training program.
- Ensure training courses prepared at the organization level are developed and maintained according to organizational standards.

Work Product, continued:

$ Ensure a wavier procedure for required training is established and used to determine whether individuals already possess the knowledge and skills required to perform in their designated area.

$ Ensure measurements are made and used to determine the status of the training program activities.

$ Ensure training program activities are reviewed with senior management on a periodic basis.

$ Ensure the training program is independently evaluated on a periodic basis for consistency with, and relevance to, the organization's needs.

$ Ensure the training program activities and work products are reviewed and/or audited and the results are reported.

$ Ensure training records are properly maintained.

Work Product: Prepare a draft Training Plan that describes the Training Program and at a minimum addresses the following issues.

- Identifies personnel to be trained. Review the list of trainees with the system owner and users to ensure that all personnel who should receive training have been identified.
- Defines the overall approach to training and the required training courses.
- Establishes the scope of the training needed for users, management, operations, and maintenance personnel.
- Define how and when training will be conducted. Specify instructor qualifications, learning objectives, and mastery or certification requirements (if applicable).
- Identify any skill areas for which certification is necessary or desirable. Tailor the training to the certification requirements.
- Establish a preliminary schedule for the training courses. The schedule must reflect training requirements and constraints outside the project. Schedule individual courses to accommodate personnel who may require training in more than one area. Identify critical paths in the training schedule such as the time period for the product's installation and conversion to production status.

Work Product, continued:

- Define the required course(s), outline their content and sequence, and establish training milestones to meet transition schedules.
- Tailor the instruction methods to the type of material being presented. Include classroom presentation, interactive computer-assisted instruction, demonstrations, individual video presentations, and hands-on experience, either live or simulated.
- Identify trainers who are technically knowledgeable and were involved in the design and development of the system. For projects with extensive and formal training requirements, it may be necessary to provide training for the trainers.
- Consider availability of the following: users, system-tested software, training rooms and equipment, and the completion of system documentation and training materials.

Place a copy of the draft Training Plan in the Project File. The plan will be reviewed and updated during the Integration and Testing Stage.

Review Process: Conduct a structured walkthrough to assure that the draft Training Plan is accurate and complete.

Resource: A Training Plan template is available on the Software Quality & Systems Engineering web site.

Chapter: **8.0 Integration and Testing Stage**

Description: Integration and testing activities focus on interfaces between and among components of the product, such as functional correctness, system stability, overall system operability, system security, privacy and sensitive information control, and system performance requirements (e.g., reliability, maintainability, and availability). Integration and testing performed incrementally provides feedback on quality, errors, and design weaknesses early in the integration process.

In this stage, components are integrated and tested to determine whether the product meets predetermined functionality, performance, quality, interface, and security requirements. Once the product is fully integrated, system testing is conducted to validate that the product will operate in its intended environment, satisfies all user requirements, and is supported with complete and accurate operating documentation.

Input: The following items provide input to this stage.

- Project File
- Acceptance Test Plan *(draft)*
- Acquisition Plan
- Installation Plan *(draft)*
- Software modules
- Requirements Traceability Matrix *(expanded)*
- Project Test File
- Development baselines
- Transition Plan
- Operating Documentation *(draft)*
  - Users Manual
  - Developer's Reference Manual
- Training Plan *(draft)*
- Integration Test Plan
- System Test Plan
- Project Plan
- Quality Assurance Plan

High-Level Activities: The remainder of this chapter is divided into sections that describe specific high-level activities performed during this stage. These activities represent the minimum requirements for a large information systems engineering effort. *Notes* are provided, as applicable, to assist in customizing these lifecycle stage

High-Level Activities, continued:

requirements to accommodate the different sizes of information systems engineering efforts. The high-level activities are presented in the sections listed below.

8.1 Conduct Integration Testing
8.2 Conduct System Testing
8.3 Initiate Acceptance Process
8.4 Conduct Acceptance Test Team Training
8.5 Develop Maintenance Plan

Output:

Several work products are produced during this stage. The work products listed below are the minimum requirements for a large project. Deviations in the content and delivery of these work products are determined by the size and complexity of the project. Explanations of the work products are provided under the applicable activities described in the remainder of this chapter.

- Integration Test Reports
- System Test Report
- Operating Documents *(final)*
  - Users Manual
  - Developer's Reference Manual
- Training Plan *(final)*
- Installation Plan *(final)*
- Acceptance Test Plan *(final)*
- Preacceptance Checklist
- Security Checklist
- Error Reporting and Tracking System *(optional)*
- Requirements Traceability Matrix *(final)*
- Maintenance Plan *(draft)*
- Project Plan *(revised)*

A matrix showing the work products associated with each high-level activity is provided in *Exhibit 8.0-1, Integration and Testing Stage Activities and Work Products by Project Size*. The matrix also shows which work products are deliverables and whether they are required or optional for small, medium, and large projects.

Review the Project Plan for accuracy and completeness of all Integration and Testing Stage activities and make any changes needed to update the information.

Review Process: Quality reviews are necessary during this stage to validate the product and associated work products. The activities that are appropriate for quality reviews are identified in this chapter and Chapter 2.0, Lifecycle Model. The time and resources needed to conduct the quality reviews should be reflected in the project resources, schedule, and work breakdown structure.

Structured Walkthrough
Requirements for a peer review or a more formal structured walkthrough, are documented under *Review Process*, at the end of each Task, Subtask, or Activity section in this stage.

In-Stage Assessment
Schedule at least one ISA prior to the Integration and Testing Stage Exit process. Additional ISAs can be performed during the stage, as appropriate. Periodic reviews of the integration and system test results and logs are recommended.

Stage Exit
Schedule a Stage Exit as the last activity of the Integration and Testing Stage to enable the project approvers to review project deliverables and provide a concur/non-concur position to the project manager.

References: Chapter 2.0, Lifecycle Model, *Quality Reviews* provides an overview of the Quality Reviews to be conducted on a project.

Chapter 10.0, *Maintenance* provides an overview of the maintenance process.

Conducting Structured Walkthroughs provides a procedure and sample forms that can be used for structured walkthroughs.

In-Stage Assessment Process Guide provides a procedure and sample report form that can be used for in-stage assessments.

Stage Exit Process Guide provides a procedure and sample report form that can be used for stage exits.

Resource: DOE Software Quality & Systems Engineering web site.

Bibliography: The following materials were used in the preparation of the Integration and Testing Stage chapter.

1. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Lifecycle Processes,* IEEE Std 1074-1991, New York, 1992.

2. U.S. Department of Commerce, National Institute of Standards and Technology, *Guide to Software Acceptance,* 500-180, Washington, D.C., 1990.

Exhibit 8.0-1. Integration and Testing Stage Activities and Work Products by Project Size

Work Activity		Project Size L	M	S	Work Product	Deliverables L	M	S
8.1	Conduct Integration Testing	R	R	R	Integration Test Materials	N	N	N
					Integration Test Report	R	R	R
					Requirements Traceability Matrix (*expanded*)	R	R	R
8.2	Conduct System Testing	R	R	R	System Test Materials	N	N	N
					System Test Report	R	R	R
					Operating Documents *(final)*	R	R	R
					Training Plan *(final)*	R	R	R
					Installation Plan *(final)*	R	R	A
8.3	Initiate Acceptance Process	R	R	R	Acceptance Test Plan *(final)*	R	R	R
					Preacceptance Checklist	R	R	R
					Security Checklist	A	A	A
					Error Reporting and Tracking System *(optional)*	O	O	O
					Requirements Traceability Matrix (*final*)	R	R	R
8.4	Conduct Acceptance Test Team Training	A	A	A	Training Materials	N	N	N
8.5	Develop Maintenance Plan	R	R	R	Maintenance Plan *(draft)*	R	R	R
3.8	Revise Project Plan	R	R	R	Project Plan *(revised)*	R	R	R
2.5	Conduct Structured Walkthroughs	R	R	A	Structured Walkthrough Management Summary Report	N	N	N
2.5	Conduct In-Stage Assessment	R	R	A	ISA Report Form[1]	N	N	N
2.5	Conduct Integration and Testing Stage Exit	R	R	A	Stage Exit Meeting Summary	N	N	N

Size: L = Large, M = Medium, S = Small

Minimum Requirements: R = Required, A = As Appropriate, N = Not a Deliverable[2]

O = Optional work product

[1] = Completed by reviewer

[2]A deliverable is a work product that is identified in the Project Plan and is formally delivered to the system owner and other project stakeholders for review and approval.

Activity: **8.1 Conduct Integration Testing**

Responsibility: Project Team Developers

Description: During integration, the components constructed by the development staff, off-the-shelf software purchased from vendors, and reusable code or modules obtained from other sources are assembled into one product. Each assembly is tested in a systematic manner in accordance with the Integration Test Plan. An incremental approach to integration enables verification that as each new component is integrated, it continues to function as designed and both the component and the integrated product satisfy their assigned requirements.

Integration testing is a formal procedure that must be carefully planned and coordinated with the completion dates of the unit-tested modules. Integration testing begins with a structure where called sub-elements are simulated by stubs. A stub is a simplified program or dummy module designed to provide the response (or one of the responses) that would be provided by the real sub-element. A stub allows testing of calling program control and interface correctness. Stubs are replaced by unit-tested modules or builds as integration testing proceeds. This process continues one element at a time until the entire system has been integrated and tested.

Integration testing may be performed using "bottom up" or "top down" techniques. Most integration test plans make use of both bottom-up and top-down techniques. Scheduling constraints and the need for parallel testing will affect the test approach.

The bottom-up approach incorporates one or more modules into a build; tests the build; and then integrates the build into the structure. The build normally comprises a set of modules that perform a major function of the system. Initially, the function may be represented by a stub that is replaced when the build is integrated.

In the top-down approach, individual stubs are replaced so that the top-level control is tested first, followed by stub replacements that move downward in the structure. Using top-down integration, all modules that comprise a major function are integrated, thereby allowing an operational function to be demonstrated prior to completion of the entire system.

Work Products: Each requirement identified in the Requirements Specification must be tested during integration testing. This traceability ensures that the product will satisfy all of the requirements and will not include inappropriate or extraneous functionality. Expand the Requirements Traceability Matrix developed in the Requirements Definition Stage to relate the integration test to the requirements. Place a copy of the expanded matrix in the Project File.

At the completion of each level of integration testing, a test report is written. The report documents test results and lists any discrepancies that must be resolved before the tested components can be used as the foundation for another integration level. Place a copy of all integration test materials in the Project Test File.

A final test report is generated at the completion of integration testing indicating any unresolved difficulties that require management attention. Place a copy of the final Integration Test Report in the Project File.

Optional Work Product: A formal reporting system by which detected errors and discrepancies are recorded and fully described is recommended. These reports will help to confirm that all known errors are fixed before delivery of the completed product. Error reports also help to trace multiple instances of the same error or anomalous behavior, so that error correction and prevention assignments can be implemented. The Quality Assurance representative assigned to the project can provide assistance in developing and using an error reporting/tracking system.

Review Process: Conduct a structured walkthrough of the Requirements Traceability Matrix and final Integration Test Report.

Activity: **8.2 Conduct System Testing**

Responsibility: Project Team or Independent Test Team

Description: During system testing, the completely integrated product is tested to validate that the product meets all requirements. System response timing, memory, performance, security, and the functional accuracy of logic and numerical calculations are verified under both normal and high-load conditions. Query and report capabilities are exercised and validated. All operating documents are verified for completeness and accuracy.

System testing is conducted on the system test bed using the methodology and test cases described in the System Test Plan. The system test bed should be as close as possible to the actual production system. Either the project team or an independent test team conducts system testing to assure that the system performs as expected and that each function executes without error. The results of each test are recorded and upon completion included as part of the project test documentation.

When errors are discovered, they should be reviewed by the test team leader to determine the severity and necessary subsequent action. If appropriate, minor problems can be corrected and regression tested by the project team developers within the time frame allotted for the system test. Any corrections or changes to the product must be controlled under configuration management. Major problems may be cause to suspend or terminate the system test, which should then be rescheduled to begin after all of the problems are resolved.

Users may be encouraged to participate in the system tests to gain their confidence in the product and to receive an early indication of any problems from the user's perspective. Inform users that errors and discrepancies may occur during testing and explain the error correction, configuration management, and retest processes.

At the successful conclusion of system testing, the product is ready for installation and acceptance testing.

Work Products: Review the draft versions of the operating documents, Training Plan, and Installation Plan. Update the documents as needed. Deliver the final versions of the operating documents, Training Plan, and Installation Plan to the system owner and user for review and approval. Place a copy of the approved documents in the Project File.

Work Products, continued:

Place a copy of all system test materials (e.g., inputs, outputs, results, and error logs) in the Project Test File.

Generate a test report at the conclusion of the system test process. The report documents the system test results and lists any discrepancies that must be resolved before the software product is installed and prepared for acceptance testing. Place a copy of the report in the Project File.

Review Process: Conduct a structured walkthrough of the system test report.

Activity: **8.3 Initiate Acceptance Process**

Responsibility: Project Manager

Description: The acceptance process is used to officially accept new or modified products that satisfy the project requirements and are fully operational. The initiation of the acceptance process begins after the successful completion of system testing. Prior to the initiation of the acceptance process, review the draft Acceptance Test Plan. Make any additions or changes needed to assure that the test plan reflects the current version of the requirements.

The acceptance process is initiated with the completion of a Pre-acceptance Checklist. This list helps to ensure that all necessary pre-acceptance activities have been completed and that the required operating documents were developed and approved. The Pre-acceptance Security Issues Checklist helps to ensure that security issues were addressed and should be completed by the system owner, as appropriate.

As part of the acceptance process, ensure that all documented issues identified during previous quality reviews have been resolved. Also, the support team should have been identified, including help-desk support. Copies of the approved operating and project documents should be provided to the support team who will maintain the system once it is accepted and transitioned to operational status. An operational analysis of the project should be conducted for support issues.

Procedure: To initiate the acceptance process, the project manager completes the Pre-acceptance Checklist and the Pre-acceptance Security Issues Checklist, obtains the concurrence signature of the system owner (if required), and schedules an initial acceptance process meeting. More than one meeting may be necessary to accommodate users at different locations or with varying requirements.

Work Product: Each requirement identified in the Requirements Specification must be tested during acceptance testing. This traceability ensures that the product has satisfied all of the requirements and does not include inappropriate or extraneous functionality. Expand and finalize the Requirements Traceability Matrix developed in the Requirements Definition Stage to relate the acceptance test to the requirements. Place a copy of the expanded matrix in the Project File.

Review the draft version of the Acceptance Test Plan, and update as needed. Deliver the final version of the Acceptance Test Plan to the system owner, user, and other project stakeholders for review and approval prior to conducting any acceptance tests. Place a copy of the approved Acceptance Test Plan in the Project File.

After the Pre-acceptance Checklist and Pre-acceptance Security Issues Checklist are reviewed by the Quality Assurance representative supporting the project, place copies in the Project File.

Review Process: Conduct structured walkthroughs of the Requirements Traceability Matrix, Pre-acceptance Checklist and Pre-acceptance Security Issues Checklist.

Resource: Samples of the Pre-acceptance Checklist and Pre-acceptance Security Issues Checklist are available on the Software Quality & Systems Engineering web site.

Activity: **8.4 Conduct Acceptance Test Team Training**

Responsibility: Project Team

Description: If the project team is not conducting the Acceptance Test, training may be required for the personnel performing the testing. The acceptance test participants and their experience with the product and the operating environment should have been identified in the Acceptance Test Plan.

The level of training will depend on the testers' familiarity with the product and the platform on which the product will run. The advantage of having users acceptance test the product is that they are the experts most familiar with the business information flow and how the product must fit into the workplace.

It is recommended that the operating documents and other test materials be distributed to the test team prior to the actual start of the acceptance test training. This will give the test team time to become familiar with the product and the test process and procedures.

Activity: **8.5 Develop Maintenance Plan**

Responsibility: Project Manager

Description: The purpose of the Maintenance Plan is to determine the scope of the maintenance effort, identify the process and tools, quantify the maintenance effort (personnel and resources), and identify anticipated maintenance requirements. The Maintenance Plan needs to define the maintenance process and its boundaries or scope. The maintenance process beginning point should be defined (e.g., receipt of a change request or planned COTS version upgrade) and the ending action should be defined (e.g., delivery and sign-off of a product). The process is a natural outgrowth of many of the configuration management procedures. A description of the overall flow of work within the maintenance process should be included. The maintenance process can be tailored to the type of maintenance being performed and can be divided in several different ways. This can include different processes for corrections or enhancements or small or large changes.

The maintenance requirements need to be identified and quantified (sized) in the Maintenance Plan to determine the future maintenance load for the organization. The following issues should be considered when defining the requirements.

- Expected external or regulatory changes to the product
- Expected internal changes to support new requirements
- Requirements deferred from current project to later release
- Wish-list of new functions and features
- Expected upgrades for performance, adaptability, or connectivity.
- New lines of business that need to be supported
- New technologies that need to be incorporated

The requirements for the maintenance staff also need to be established. At this stage, the maintenance plan should address the following:

- Number of maintainers, their job descriptions, and required skill levels
- Experience level of the maintenance staff
- Documented maintenance processes at the systems and program levels
- Actual methods used by development staff
- Tools used to support the maintenance process
- Current work load and estimates of future needs

Description, continued:

An important part of the maintenance plan is an analysis of the hardware and software most appropriate to support the maintenance organization's needs. The development, maintenance, and test platforms should be defined and differences between the environments described. Tools sets that enhance productivity should be identified and provided. Tools should be accessible to all who need them, and sufficient training provided so that their use will be well understood.

Although all systems need maintenance, there comes a time when maintenance is no longer technically or fiscally viable. Issues such as resources, funds, and priorities may dictate that a system should be replaced rather than changed. The maintenance plan should identify the criteria that indicate the product is ready for retirement or replacement, such as the failure rate, age of code, and incompatibility with current technology. *Computer Systems Retirement Guidelines*, available on the Software Quality & Systems Engineering web site, provides a process and sample forms that can be used for retiring a system.

Work Product:

Develop the Maintenance Plan. Place a copy of the draft Maintenance Plan in the Project File. As part of the Systems Installation and Acceptance stage, once system installation and acceptance are complete, determine if the Maintenance Plan needs to be revised, if so, update and finalize the Maintenance Plan. Place a copy of the final Maintenance Plan in the Project File.

Review Process:

Conduct a structured walkthrough to ensure that the Maintenance Plan accurately reflects the necessary information.

The Maintenance Plan is formally reviewed during the In-Stage Assessment and Stage Exit processes.

Reference:

Refer to Chapter 10, *Maintenance*, for more information on maintaining the product.

Resource:

A template for the Maintenance Plan is available on the Software Quality & Systems Engineering web site.

Chapter: **9.0 Installation and Acceptance Stage**

Description: Installation and acceptance of the product are initiated after the system test has been successfully completed. This stage involves the activities required to install the software, databases, or data that comprise the product onto the hardware platform at the site(s) of operation. The objectives of the activities in this stage are to verify that the product meets design requirements and to obtain the system owner's acceptance and approval of the product. The activities associated with this stage should be performed each time the product is installed at an acceptance test site or production site.

User training may be required to complete the installation process. A description of the training necessary for developers, testers, users, and operations staff is provided in the Training Plan.

Input: The following items provide input to this stage:

- $ Integration Test Materials
- $ System Test Materials
- $ Operating Documents
  - Users Manual
  - Developer's Reference Manual
- $ Training Plan
- $ Installation Plan
- $ Conversion Plan
- $ Acceptance Test Plan
- $ Preacceptance Checklist
- $ Security Checklist
- $ Maintenance Plan (*draft*)
- $ Project Plan *(revised)*
- $ Quality Assurance Plan

High-Level Activities: The remainder of this chapter is divided into sections that describe the specific high-level activities performed during this stage. These activities represent the minimum requirements for a large information systems engineering effort. *Notes* are provided, as applicable, to assist in customizing these lifecycle stage requirements to accommodate the different sizes of information systems engineering efforts. The high-level activities are presented in the sections listed below.

9.1 Perform Installation Activities
9.2 Conduct Installation Tests
9.3 Conduct User Training
9.4 Conduct Acceptance Test
9.5 Conclude Acceptance Process
9.6 Transition to Operational Status

Output: Several work products are produced during this stage. The work products listed below are the minimum requirements for a large project. Deviations in the content and delivery of these work products are determined by the size and complexity of the project. Explanations of the work products are provided under the applicable activities described in the remainder of this chapter.

- $ Converted data or system files
- $ Installation Test materials
- $ User Training materials
- $ Acceptance Test Report
- $ Acceptance Checklist
- $ Operational software product
- $ Operating documents
- $ Maintenance Plan *(final)*
- $ Project Plan *(revised)*

A matrix showing the work products associated with each high-level activity is provided in *Exhibit 9.0-1, Installation and Acceptance Stage Activities and Work Products*. The matrix also shows which work products are deliverables and whether they are required or optional for small, medium, and large projects.

Review the Project Plan for accuracy and completeness of all Installation and Acceptance Stage activities and make any changes needed to update the information.

Review Process: Quality reviews are necessary during this stage to validate the product and associated work products. The activities that are appropriate for quality reviews are identified in this chapter and Chapter 2.0, Lifecycle Model. The time and resources needed to conduct the quality reviews should be reflected in the project resources, schedule, and work breakdown structure.

Structured Walkthrough
Requirements for a peer review or a more formal structured walkthrough, are documented under *Review Process*, at the end of each Task, Subtask, or Activity section in this stage.

In-Stage Assessment
Schedule at least one ISA prior to the Installation and Acceptance Stage Exit process. Additional ISAs can be performed during the stage, as appropriate.

Stage Exit
Schedule a Stage Exit as the last activity of the Installation and Acceptance Stage to enable the project approvers to review project deliverables and provide a concur/non-concur position to the project manager.

References: Chapter 2.0, Lifecycle Model, *Quality Reviews* provides an overview of the Quality Reviews to be conducted on a project.

Conducting Structured Walkthroughs provides a procedure and sample forms that can be used for structured walkthroughs.

In-Stage Assessment Process Guide provides a procedure and sample report form that can be used for in-stage assessments.

Stage Exit Process Guide provides a procedure and sample report form that can be used for stage exits.

Resource: DOE Software Quality & Systems Engineering web site.

Bibliography: The following materials were used in the preparation of the Installation and Acceptance Stage chapter.

1. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Lifecycle Processes,* IEEE Std 1074-1991, New York, 1992.

2. U.S. Department of Commerce, National Institute of Standards and Technology, *Guide to Software Acceptance,* 500-180, Washington, D.C., 1990.

3. U.S. Department of Energy, *Automated Data Processing Systems Development Methodology, Volume 1,* K/CSD/INF/86-3, Vol.1, R3, prepared under contract by Martin Marietta Energy Systems, Inc, at Oak Ridge National Laboratory, August 1987.

4. U.S. Department of Energy, *DOE/NV Software Management Plan,* Nevada Operations Office, May 1991.

Exhibit 9.0-1. Installation and Acceptance Stage Activities and Work Products by Project Size

Work Activity		Project Size L	M	S	Work Product	Deliverables L	M	S
9.1	Perform Installation Activities	R	R	R	Converted Data or System Files	I	I	I
9.2	Conduct Installation Tests	R	R	R	Installation Test Materials	R	R	R
9.3	Conduct User Training	R	R	R	User Training Materials	R	R	R
9.4	Conduct Acceptance Test	R	R	R	Acceptance Test Materials Acceptance Test Report	R R	R R	R R
9.5	Conclude Acceptance Process	R	R	A	Acceptance Checklist	R	R	A
9.6	Transition to Operational Status	R	R	R	Operational Software Product Operating Documents	R R	R R	R R
8.5	Revise Maintenance Plan	R	R	R	Maintenance Plan (final)	R	R	R
3.8	Revise Project Plan	R	R	R	Project Plan (revised)	R	R	R
2.5	Conduct Structured Walkthrough(s)	R	R	A	Structured Walkthrough Management Summary Report	N	N	N
2.5	Conduct In-Stage Assessment	R	R	A	ISA Report Form[1]	N	N	N
2.5	Conduct Installation and Acceptance Stage Exit	R	R	A	Stage Exit Meeting Summary	N	N	N

Size: L = Large, M = Medium, S = Small

Minimum Requirements: R = Required, A = As Appropriate, N = Not a Deliverable[2], I = Input to another Deliverable

I = Input to other deliverables

[1] = Completed by reviewer

[2]A deliverable is a work product that is identified in the Project Plan and is formally delivered to the system owner and other project stakeholders for review and approval.

Activity: **9.1 Perform Installation Activities**

Responsibility: Project Team

Description: The installation process involves loading, copying, or migrating the software and data, if required, to the production platform and the provision of operating documentation and other support materials at each site. The installation of firmware, hardware, and communications equipment may also be involved.

If a current system exists, implement system and data conversion in accordance with the procedures described in the Conversion Plan. Each data and file conversion should include a confirmation of data and file integrity. Determine what the output in the new product should be compared with the current system, and assure that the data and files are synchronized.

At each installation site, inspect the facility to assure that site preparation is complete and in accordance with the Installation Plan. Initiate any actions that are needed to complete the preparations. Conduct an inventory of all vendor provided hardware, software, firmware, and communications equipment in accordance with the Acquisition Plan.

Follow the procedure specified in the Installation Plan when installing the software, hardware, and other equipment. Monitor all installation activities including those performed by vendors.

Procedure: Use the following procedure to perform the installation activities.

- Coordinate the installation with the system owner, users, operations staff, and other affected organizations.
- Ensure that any necessary modifications to the physical installation environment are completed.
- Inventory and test the hardware that will support the product. This inventory should be performed in advance of the planned installation date to allow time for missing hardware to be obtained and malfunctioning equipment to be replaced or repaired.
- If the product requires an initial data load or data conversion, install and execute the tested programs to perform this process.
- Install the software product onto the hardware platform.

This page intentionally left blank.

Activity: **9.2 Conduct Installation Tests**

Responsibility: Project Team

Description: Ensure the integrity and quality of the installed product by executing the installation tests defined in the Installation Plan. Testing is performed to verify that the product has been properly installed and is fully operational.

The installation test(s) are designed to validate all functions of the product and should specify a standard set of test results and tolerances. If the product being installed is a modification to an existing system, all remaining functions and code that may be affected by the new product should be tested.

Document any problems and identify corrective action. Select a diagnostic package that will pinpoint problems quickly and allow for timely corrections. Retest all equipment and software after a repair, replacement, or modification.

Certify each component on successful completion of installation and checkout. When installation is complete, rerun a portion or all of the system test and dry-run the acceptance test procedures to verify correct operation of the product.

Conduct installation testing to verify the following:

- Security functions
- Integration with the current software
- Interfaces with other systems
- System functionality based on the requirements

Work Product: Place a copy of all Installation Test materials in the Project File.

Review Process: Conduct structured walkthroughs of the Installation Test materials.

Activity: **9.3 Conduct User Training**

Responsibility: Project Team

Description: User training is an important factor in the success of the operational product. During training, most users will receive their first hands-on experience with the product. Operations and maintenance staff may also be trained to use, monitor, and maintain the product. The objective of the training is to provide the trainee with the basic skills needed to effectively use the product and to raise the user's confidence and satisfaction with the product.

The type of training will depend on the complexity of the product, and the number and location of the users to be trained. Alternative training formats include formal classroom training, one-on-one training, computer-based instruction, and sophisticated help screens and online documentation. Conduct the training as described in the Training Plan.

Consider conducting a pilot test of the training session(s). Include members of the project team, the system owner, and key users. Have all participants evaluate the training content, instruction, and support materials. Make any necessary changes to the training program prior to general user training sessions.

If consecutive training classes are conducted, feedback should be requested from the participants and used to continuously improve the training approach, methods, and materials.

At the completion of the training, users should have a thorough understanding of the input requirements of each transaction, the processing that takes place, and the types of output that are generated.

Work Product: Submit a copy of the training materials to the system owner and user for review and approval. Place a copy of the approved training materials in the Project File. Training materials are subject to the same configuration control procedures as the other operating documents and should remain current with changes to the product.

Review Process: Conduct structured walkthroughs of the training materials. The pilot test of the training session(s) helps ensures delivery of a quality training program.

Activity: **9.4 Conduct Acceptance Test**

Responsibility: Acceptance Test Team

Description: Acceptance of a delivered product is the ultimate objective of a development project. Acceptance testing is used to demonstrate the product's compliance with the system owner's requirements and acceptance criteria.

At the system owner's discretion, acceptance testing may be performed by the project team, by the system owner and users with support from the project team, or by an independent verification and validation team. Whenever possible, users should participate in acceptance testing to assure that the product meets the users' needs and expectations. All acceptance test activities should be coordinated with the system owner, user(s), operations staff, and other affected organizations.

Acceptance testing is conducted in the production environment using acceptance test data and test procedures established in the Acceptance Test Plan. Testing is designed to determine whether the product meets functional, performance, and operational requirements. If acceptance testing is conducted on an incremental release basis, the testing for each release should focus on the capabilities of the new release while verifying the correct operation of the requirements incorporated in the previous release.

Acceptance testing usually covers the same requirements as the system test. Acceptance testing may cover additional requirements that are unique to the operational environment. The results of each test should be recorded and included as part of the project test documentation.

Subject the test environment to strict, formal configuration control to maintain the stability of the test environment and to assure the validity of all tests. Review the acceptance test environment, including the test procedures and their sequence, with the system owner and user before starting any tests.

Testing is complete when all tests have been executed correctly. If one or more tests fail, problems are documented, corrected, and retested. If the failure is significant, the acceptance test process may be halted until the problem is corrected.

Work Product: Prepare a formal Acceptance Test Report. Summarize the test procedures executed, any problems detected and corrected, and the projected schedule for correcting any open problem reports. Place a copy of all acceptance test materials in the Project Test File.

Review Process: Conduct an Operational Readiness Review at the completion of acceptance testing. This review is a combined quality assurance and configuration management activity. It focuses on the results of the acceptance test and the readiness of the product to go into production including review of security, sensitive information and privacy control.

The Operational Readiness Review includes a functional configuration audit to determine whether the test records demonstrate that the product meets its technical requirements, and a physical configuration audit to determine whether the product technical documentation is complete and accurately describes the product.

During the Operational Readiness Review examine acceptance test results with the system owner and user. Document any problems, determine solutions to the problems, and implement action plans. Once any problems associated with the acceptance test are resolved, the product is ready for formal acceptance by the system owner.

A successful Operational Readiness Review establishes the operational baseline for the product. The operational baseline is the final baseline. It consists of the product and the technical documentation that describes the operational product and its characteristics. It contains the current functional baseline, the product baselines for the configuration items comprising the system, and other system-level technical documentation generated during the lifecycle.

If the operational product requires enhancements or changes to correct problems, each new release should be preceded by an Operational Readiness Review, after which the updated system documentation is established as a new operational baseline superseding the previous one.

Activity: **9.5 Conclude Acceptance Process**

Responsibility: Project Manager

Description: The acceptance process is used to officially accept new or modified products that satisfy the users' requirements and are fully operational. The acceptance process is concluded when the acceptance test has been successfully completed, the product has been installed and is operational at all user sites, and complete operating documentation describing the product has been approved and delivered.

At the conclusion of the acceptance process, responsibility for the product is formally transferred from the project team to the system owner and maintenance staff.

Procedure: To conclude the acceptance process the project manager completes the Acceptance Checklist, obtains the concurrence signature of the system owner (if required), and schedules an acceptance meeting. More than one meeting may be necessary to accommodate users at different locations or with varying requirements.

Work Product: The Acceptance Checklist is completed and submitted to the Quality Assurance representative supporting the project. This list helps to ensure that all necessary acceptance activities have been completed and that the required operating documents were developed and approved.

A formal written acceptance of the product is produced by the system owner to verify that the product is acceptable and ready for production.

Review Process: Conduct structured walkthroughs or peer reviews of the Acceptance Checklist.

Resource: A sample of an acceptance checklist is available on the Software Quality & Systems Engineering web site.

Activity: **9.6 Transition to Operational Status**

Responsibility: Transition Team

Description: The transition of the product to full operational status begins after the formal acceptance by the system owner. Use the procedures described in the Transition Plan to implement the transition processes. Conduct or support stress tests and other operational tests. Determine product tolerances to adverse conditions, failure modes, recovery methods, and specification margins. Complete any training and certification activities. Ensure that support to be provided by contractors begins as planned.

The project team is usually expected to provide operational and technical support during the transition. Identify project team personnel with a comprehensive understanding of the product who can provide assistance in the areas of installation and maintenance, test, and documentation of changes. Technical support may involve the analysis of problems in components and operational procedures, the analysis of potential enhancements, and vendor-supplied upgrades to components (such as the operating system or database management system).

Transition to full operational status should be an event-oriented process that is not complete until all transition activities have been successfully performed. Withdraw the support of the project team personnel in a gradual sequence to ensure the smooth operation of, and user confidence in, the product. At the conclusion of the transition process, plan a formal transfer of all responsibility to the maintenance staff.

All Project File materials, operating documents, a list of any planned enhancements, and other pertinent records should be turned over to the maintenance staff. Access rules must be modified to provide access to the product by the maintenance staff and to remove access by the project team and other temporary user accesses. Programs, files, and other support software should be in the production library and deleted from the test library, where appropriate.

For major systems involving multiple organizations and interfaces with other systems, a formal announcement of the transition to production is recommended. The announcement should be distributed to all affected groups. The names and telephone numbers of the maintenance staff should be included.

Work Product: The system is transitioned into operational status. Project File materials, operating documents, and other pertinent records are turned over to the maintenance staff.

Review Process: All reviews related to the functionality were completed prior to the system being placed into operational status

Chapter: **10.0 Maintenance**

Description: This chapter describes an iterative process for conducting system maintenance activities. The process prescribes a minimal set of criteria that are necessary for project management and quality assurance processes, control, and management of the planning, execution, and documentation of maintenance activities. The use of automated tools to facilitate requirements definition, design, coding, and system documentation is encouraged. The selection and implementation of tools varies among sites and organizations.

The basic maintenance process model includes input, process, output, and control for maintenance. It is based on the same information systems engineering principles and preferred practices that lower risk and improve quality during the planning and development stages of the lifecycle. The process model supports the concept that planned changes should be grouped and packaged into scheduled releases that can be managed as projects. This proven approach allows the maintenance team to better plan, optimize use of resources, take advantage of economies of scale, and better control outcome in terms of both schedule and product quality.

Each organization performing system maintenance activities should have a local documented procedure for handling emergency changes that cannot be implemented as part of a scheduled release. Generally, these changes include fixes to correct defects and updates to meet unscheduled business or legal requirements. Emergency changes should also be integrated into the next release for full regression testing and documentation updates.

Stages: The activities to be performed during maintenance are grouped into logically related segments of work called "stages." These stages are similar to those referenced in the planning and development stages of the information systems development lifecycle. The stages are presented in the sections listed below.

10.1 Problem/Modification Identification Stage
10.2 Analysis Stage
10.3 Design Stage
10.4 Construction Stage
10.5 System Test Stage
10.6 Acceptance Stage
10.7 Delivery Stage

A matrix depicting the maintenance process model stages is provided in *Exhibit 10.0-1, Process Model for Maintenance.*

Note: The maintenance process model does not presume the use of any particular information systems development methodology (e.g., waterfall, spiral). This process is valid regardless of size, complexity, criticality, application of the product, or usage of the product in the system to be maintained.

The maintenance stages can be tailored (i.e., logically combining stages or outputs) as appropriate. Stages may be combined to more effectively manage the project. Decisions to combine stages are agreed to by the designated approvers during the Analysis Stage. Factors that can influence the number of stages include level of effort, complexity, visibility, and business impact. Guidance to assist with decisions to combine stages is presented in *Exhibit 10.0-2, Tailoring For Size*.

Project Management: To the extent possible, all maintenance activity should be managed as a project to gain the benefits inherent in project management and to enable tracking of activities and costs. The extent of project management activity will vary, and should be tailored according to the size, complexity, and impact of the change or enhancement.

Review Processes: In each stage, one or more structured walkthroughs are conducted to validate work products. The *Conducting Structured Walkthroughs* Process Guide provides a procedure and sample forms that can be used for structured walkthroughs.

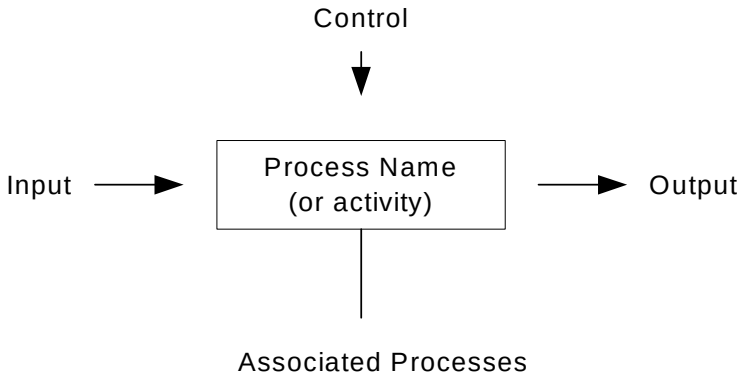
In maintenance, and especially for major modifications, one or more In-Stage Assessments are conducted as part of the quality assurance activities for each stage. This process is documented in the *In-Stage Assessment (ISA) Process Guide*.

A Stage Exit is conducted at the end of each stage of maintenance. This process, which includes definition of participant roles and the review and approval process, is documented in the *Stage Exit Process Guide*.

Metrics: Metrics/measures and associated factors for each stage should be collected and reviewed at appropriate intervals. *Exhibit 10.0-3, Process Model Metrics for Maintenance*, provides metrics for each stage of maintenance. Metrics/measures captured for maintenance should enhance the implementation and management of this process.

Conventions: The following convention is used in each exhibit depicting a maintenance stage.

Exhibit 10.0-0. Exhibit Convention



An "associated process" is one that is executed in support of system maintenance, but is itself not defined in this chapter (e.g., Stage Exit).

Bibliography: The following materials were used in the preparation of the Maintenance chapter.

1. DeMarco, Tom, *Controlling Software Projects*, New York, 1998.

2. Frame, Davidson J., *Managing Projects in Organizations*, San Francisco, 1995.

3. Frame, Davidson J., *The New Project Management*, San Francisco, 1994.

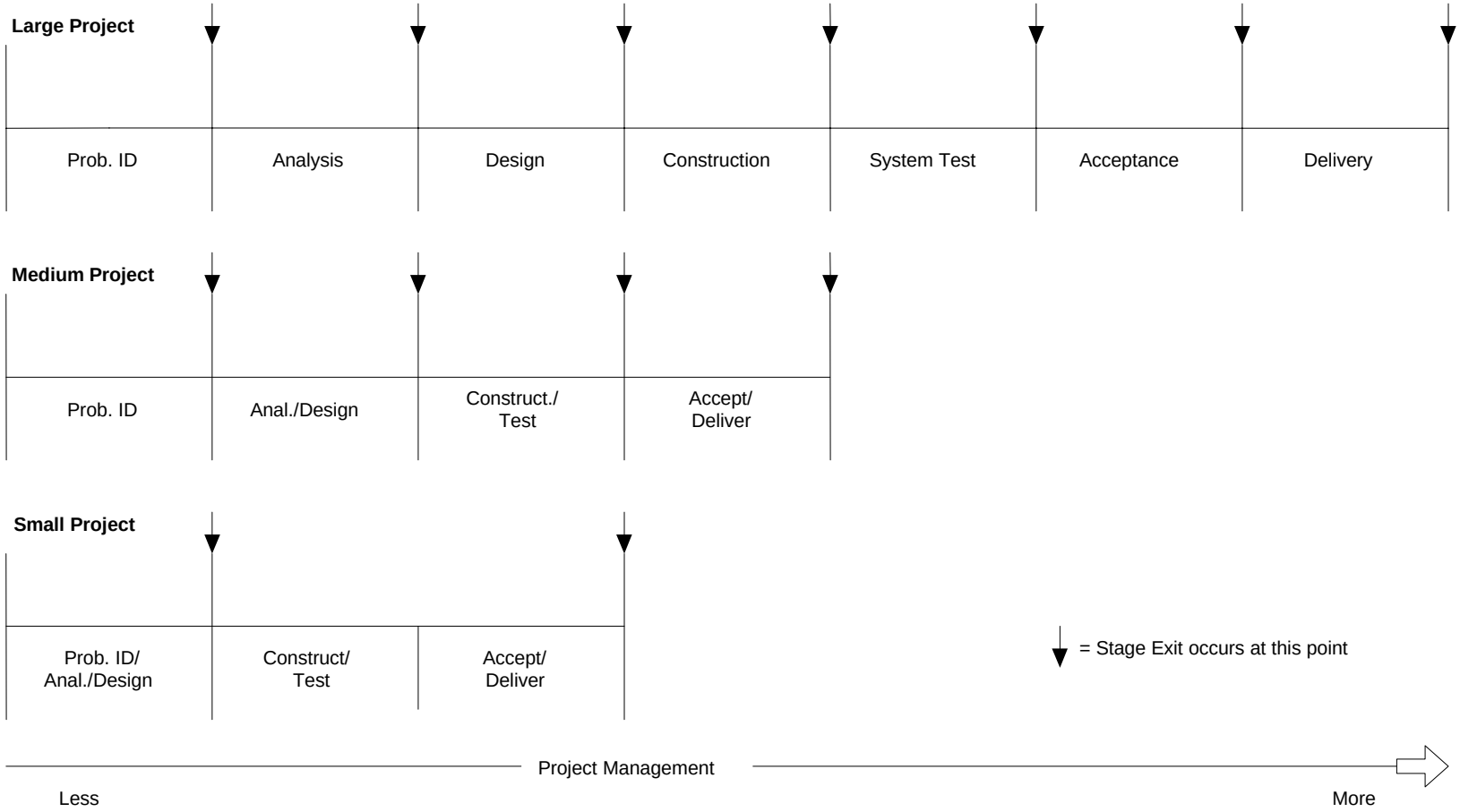
4. Page-Jones, Meilir, *Practical Project Management*, New York, 1998.

5. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Software Maintenance*, IEEE Std 1219-1992, New York, 1993.

Exhibit 10.0-1. Process Model for Maintenance

	Problem Identification Stage	Analysis Stage	Design Stage	Construction Stage	System Test Stage	Acceptance Stage	Delivery Stage
Input	Modification Request (MR)	Project/system document Project file information Validated MR	Project/system document Existing source code Database(s) Analysis stage output	Current source code Product/system document Results of design stage	Updated documentation Test Readiness Review Report Updated system Project/maintenance plan	Test Readiness Review Report Integrated system Acceptance test: Plans Cases Procedures	Tested/accepted/ baselined system
Process	Assign change number Categorize Accept or reject change Preliminary effort estimate	Feasibility analysis Detailed analysis Redocument, if needed	Revise: Requirements System doc. Module doc. Project plan Create test cases	Code Unit test Test Readiness Review Revisit project risk	Functional test Interface testing Regression testing Test Readiness Review	Acceptance test Interoperability test Functional Configuration Audit (FCA)	Physical Configuration Audit (PCA) Install Training User notification Archival system for backup
Output	Validated MR Process determinations	Feasibility Report Detailed Analysis Report Updated: Requirements Modification list Test strategy Project plan	Revised: Modification list Detailed analysis Updated: Design baseline Test plans Project plan Constraints and risks	Updated: Design documents Test documents User documents Training materials Project plan Statement of risk Test readiness review report	Tested system Test reports Updated project plan Test Readiness review report	New system baseline Acceptance Test Report FCA Report Updated project plan	PCA Report Version Description Document (VDD)
Review Assurance Approve	Structured Walkthrough(s)	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough(s) In-Stage Assessment(s) Stage Exit	Structured Walkthrough Stage Exit
Metrics	*See Exhibit 10.0-3, Process Model Metrics for Maintenance*						

Exhibit 10.0-2. Tailoring for Size



Size is used as a guide to help determine the appropriate degree of project management, and which stages may be combined for a given effort. Within this context, size is a combination of level of effort required (all activities) and complexity of the modification. Attributes of complexity include technology, team skills, interfaces, and level of understanding of requirements. Other factors that can influence tailoring include risk, visibility, and business impact.

Exhibit 10.0-3. Process Model Metrics for Maintenance

	Problem Identification Stage	Analysis Stage	Design Stage	Construction Stage	System Test Stage	Acceptance Stage	Delivery Stage
Factors	Correctness Maintainability	Flexibility Traceability Usability Reusability Maintainability Comprehensibility	Flexibility Traceability Reusability Testability Maintainability Comprehensibility Reliability	Flexibility Traceability Maintainability Comprehensibility Reliability	Flexibility Traceability Verifiability Testability Interoperability Comprehensibility Reliability	Flexibility Traceability Interoperability Testability Comprehensibility Reliability	Completeness Reliability
Metrics	No. of omissions on Modification Request (MR) No. of MR submittals No. of duplicate MRs Time expended for problem validation	Requirement changes Documentation error rates Effort per function area (e.g., SQA) Elapsed time (schedule) Error rates, by priority and type	S/W complexity Design changes Effort per function area Elapsed time Test plans and procedure changes Error rates, by priority and type Number of lines of code, added, deleted, modified, tested	Volume/ functionality (function points or lines of code) Error rates, by priority and type	Error rates, by priority and type Generated Corrected	Error rates, by priority and type Generated Corrected	Documentation changes (i.e. version description documents, training manuals, operation guidelines)

Note: The above level of metrics is a goal. Each organization responsible for maintenance activities should establish an individual plan to achieve this level over time.

Stage: **10.1 Problem/Modification Identification Stage**

Responsibility: Maintenance Team

Description: In this stage, product changes are identified, categorized, and assigned an initial priority ranking. Each request for a modification (i.e., Modification Request) is evaluated to determine its classification and handling priority. The classification should be identified from the following types of maintenance.

- Corrective - Change to a product after delivery to correct defects.
- Adaptive - Change to a product after delivery to keep it functioning properly in a changed or changing environment.
- Emergency - Unscheduled corrective maintenance required to keep a system operational.
- Scheduled - Change to a product after delivery on a routine basis to improve performance or maintainability.
- Perfective - Change to a product after delivery to improve performance or maintainability.

The need for modifications can be driven by any number of factors, including:

- Report of system malfunction.
- Mandatory changes required by new or changed federal or state law.
- New requirements to support business needs.
- Major enhancement or redesign to improve functionality or replace an obsolete system component.
- Operational system upgrades and new versions of resident software (e.g., COBOL, CICS, Oracle).

These factors should be considered when assigning a priority to the modification request. *Exhibit 10.1-1* (provided at the end of this section) summarizes the input, process, control, and output for the Problem/Modification Identification Stage of maintenance.

Input: Input to the Problem/Modification Identification Stage of maintenance is one or more Modification Requests.

Process: If a modification to the product is required, the following activities must occur within the maintenance process.

- Assign an identification number.
- Categorize the type of maintenance.
- Analyze the modification to determine whether to accept, reject, or further evaluate.
- Prioritize the modification according to the following categories:

Process, continued:

- Emergency (follow emergency change procedure and integrate into the next scheduled release or block of modifications)
- Mandatory (e.g., legal, safety, payroll)
- Required (has associated benefits; e.g., productivity gains, new business drivers)
- Nice to have (lower priority)

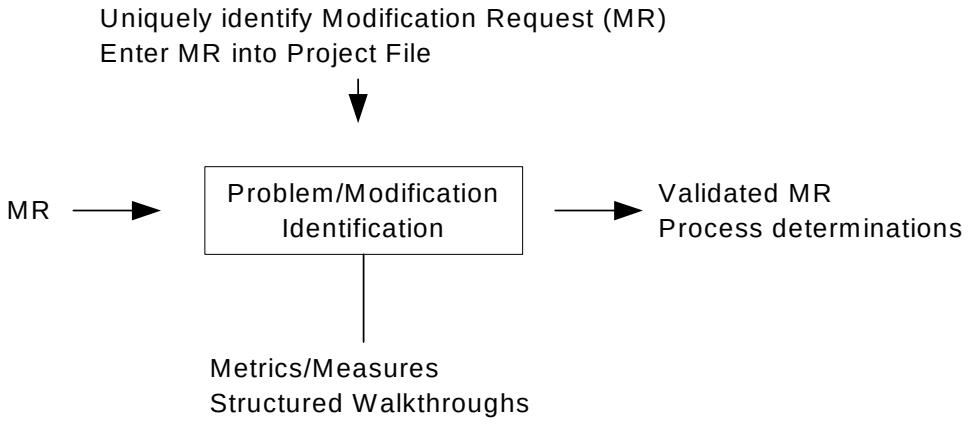
Control: Modification Requests and process determinations are uniquely identified and entered into the Project File.

Work Products: The output of this stage is the validated Modification Request and the following process determinations. Place a copy of all work products in the Project File.

- Statement of the problem or new requirement.
- Problem or requirement evaluation.
- Categorization of the type of maintenance required.
- Initial priority.
- Verification data (for corrective modifications).
- Initial estimate of resources required.

Review Process: Conduct structured walkthrough(s) as appropriate.

Exhibit 10.1-1. Problem/Modification Identification Stage



Stage: **10.2 Analysis Stage**

Responsibility: Project Team and System Owner

Description: During the Analysis Stage, the Project File information, the Modification Request(s) validated in the Problem/Modification Identification Stage, and the system and project documentation are used to study the feasibility and scope of the modification, and to develop a preliminary Project Plan for design, test, and delivery.

If the documentation is not available or is insufficient and the source code is the only reliable representation of the system, reverse engineering is recommended to ensure the overall integrity of the system. In those cases where long-lived systems have overgrown the initial base system and have poorly updated documentation, reverse engineering may be required and would evolve through the following steps:

For a smaller scope, or for local analysis on a unit level:

- Dissection of source code into formal units.
- Semantic description of formal units and declaration of functional units.
- Creation of input/output schematics of units.

For a larger scope, or for global analysis on a system level:

- Declaration and semantic description of linear flows.
- Declaration and semantic description of system applications (functions grouped).
- Creation of anatomy of the system (system architecture).

Modifications of a similar nature (i.e., affecting the same program(s)) should be grouped together whenever possible, and packaged into releases that are managed as projects. A release cycle should be established and published.

Exhibit 10.2-1 (provided at the end of this section) summarizes the input, process, control, and output for the Analysis Stage.

Input: Input to the Analysis Stage of maintenance includes the following.

- Validated Modification Request.
- Initial resource estimate and associated information.
- Project and system documentation, if available.

Process: Preliminary analysis activities include the following.

- Make a preliminary estimate of the modification size/magnitude.
- Assess the impact of the modification.
- Assign the Modification Request to a block of modifications scheduled for implementation.
- Coordinate the modifications with other ongoing maintenance tasks.

Once modifications are agreed to, grouped if appropriate, and packaged, analysis progresses and includes the following.

- Define firm requirements for the modification.
- Identify elements of the modification.
- Identify safety and security issues.
- Devise a test and implementation strategy.

In identifying the elements of the modification (creating the preliminary modification list), examine all work products (e.g., system specifications, databases, and documentation) that are affected. Each work product is identified, and generated, if necessary, specifying the portion of the product to be modified, the interfaces affected, the user-noticeable changes expected, the relative degree and kind of experience required to make changes, and the estimated time to complete the modification.

The test strategy is based on input from the previous activity identifying the elements of modification. Requirements for at least three levels of testing, including individual unit tests, integration tests, and user-oriented functional tests are defined. Regression test requirements associated with each of these levels of testing are identified as well. The test cases to be used for testing to establish the test baseline are revalidated.

Control: Control of the Analysis Stage activities includes the following.

- Retrieval of the relevant version of project and system documentation from the configuration control function of the organization.
- Review of the proposed changes and engineering analysis to assess technical and economic feasibility, and correctness.
- Identification of safety and security issues.
- Consideration of the integration of the proposed change within the existing product.
- Verification that all appropriate analysis and project documentation is updated and properly controlled.
- Verification that the test function of the organization is providing a strategy for testing the change(s), and that the change schedule can support the proposed test strategy.
- Review of resource estimates and schedules; verification of accuracy.

Control, continued:

- Technical review to select the problem reports and proposed enhancements to be implemented in the new release.

Work Products: The output of the Analysis Stage includes the following.

- Feasibility report for modification requests.
- Detailed analysis report.
- Updated requirements (including traceability list).
- Test strategy.
- Project Plan.

A written assessment, generally called a Feasibility Report, is prepared and contains the following.

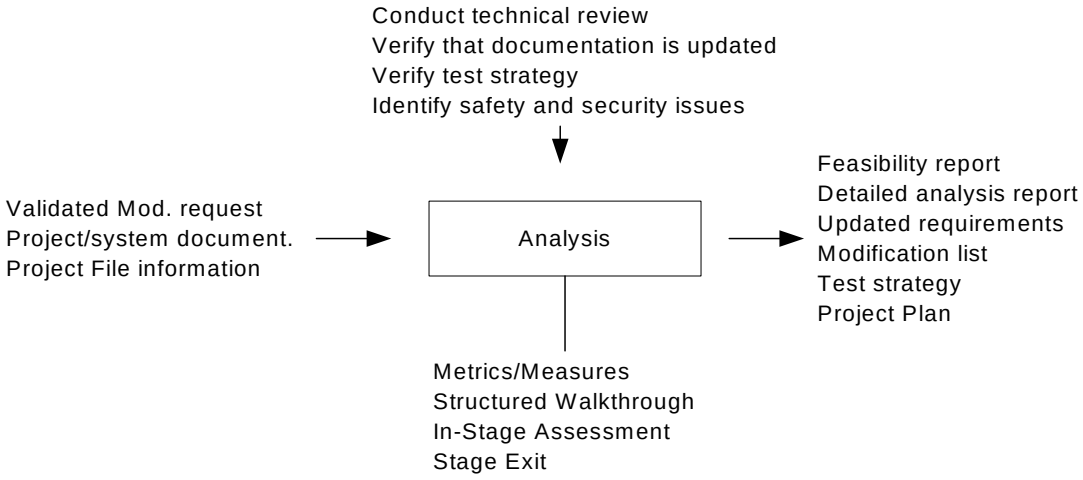
- Short and long term costs.
- The value of the benefit of making the modification (usually provided by the system owner).
- Solution approach, including prototyping if applicable.
- Safety and security implications.
- Human factors.

A project plan states how the design, implementation, testing, and delivery of the modification are to be accomplished with a minimal impact to current users.

Review Processes: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

At the end of the Analysis Stage, a risk analysis is performed. Using the output of the Analysis Stage, the preliminary resource estimate is revised, and a decision is made on whether to proceed to the Design Stage.

Exhibit 10.2-1. Analysis Stage



Stage: **10.3 Design Stage**

Responsibility: Project Team

Description: In the Design Stage, all current system and project documentation, existing software and databases, and the output of the Analysis Stage are used to design the modification to the system. *Exhibit 10.3-1* (provided at the end of this section) summarizes the input, process, and output for the Design Stage of maintenance.

Input: Input to the Design Stage of maintenance includes the following.

- Analysis Stage output, including:
  - Detailed analysis
  - Updated statement of requirements
  - Preliminary modification list (identification of affected elements)
  - Test strategy
  - Project Plan
- System and project documentation.
- Existing source code, comments, and databases.

Process: The process steps for the Design Stage include the following.

- Identify selected modules.
- Modify module documentation (e.g., data and control flow diagrams, schematics).
- Create test cases for the new design, including safety and security issues.
- Identify/create regression tests.
- Identify documentation (system/user) update requirements.
- Update modification list.
- Document any known constraints that influence the design, and any risks that have been identified. Where possible, actions, taken or recommended, that mitigate risk should also be documented.

Control: The following control activities are performed during the Design Stage of a modification.

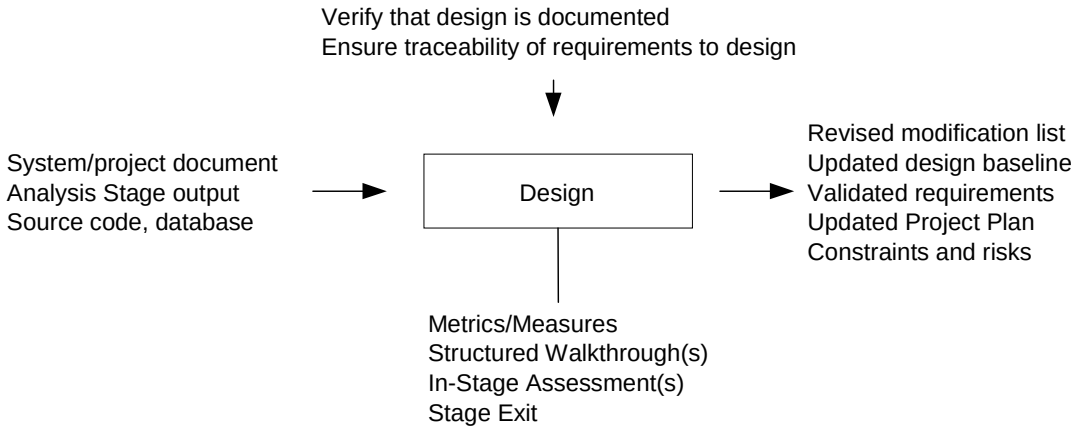
- Verify that the new design/requirement is documented as an authorized change.
- Verify the inclusion of new design material, including safety and security issues.
- Verify that the appropriate test documentation has been updated.
- Complete the traceability of the requirements to the design.

Work Products: The output of the Design Stage of maintenance includes the following.

- Revised modification list.
- Updated design baseline.
- Updated test plans.
- Revised detailed analysis.
- Verified requirements.
- Updated Project Plan.
- Documented constraints and risks.

Review Process: Conduct Structured Walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.3-1. Design Stage



Stage: **10.4 Construction Stage**

Responsibility: Project Team

Description: In the Construction Stage, the results of the Design Stage, the current source code, the project and system documentation, (i.e., the entire system as updated by the prior stages) is used to drive the construction effort. *Exhibit 10.4-1* (provided at the end of the section) summarizes the input, control, and output for the Construction stage.

Input: Input to the Construction Stage of maintenance includes the following.

- Results of the Design Stage.
- Current source code, comments, and databases.
- Project and system documentation.

Process: The Construction Stage includes the following tasks, which may be conducted in an incremental, iterative approach.

- Coding and unit testing.
- Integration.
- Revisit project risk.
- Test readiness review.

Coding and Unit Testing
Implement the change into the code and perform unit testing. Other quality assurance and verification and validation processes may be required for safety-related code. The Quality Assurance Representative can help with specific issues.

Integration
After the modifications are coded and unit tested, or at appropriate intervals during coding, the modified product is integrated with the system, and integration and regression tests are refined and performed. All effects (e.g., functional, performance, usability, safety) of the modification on the existing system are assessed and noted. A return to the coding and unit testing tasks is made to remove any unacceptable impacts.

Risk Analysis and Review
Risk analysis and review are performed periodically during the Construction Stage rather than at the end, as in the Design and Analysis Stages. Metrics/measurement data should be used to quantify risk analysis.

Test Readiness Review
To assess whether the product is ready to enter system testing, a Test Readiness Review is conducted. This is a self-assessment to determine if items including code, documentation, libraries, hardware, telecommunication lines, and schedules are ready for system test to begin on the scheduled date.

Control: Control of the Construction Stage includes the following activities.

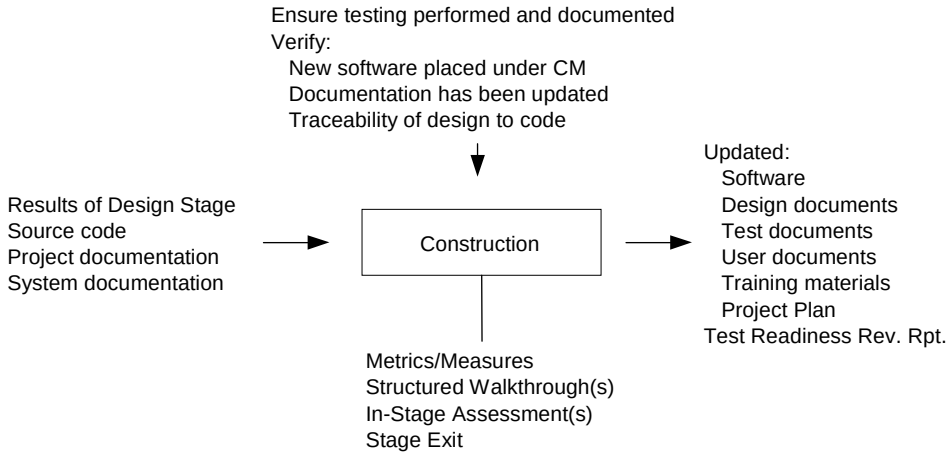
- Ensure that unit and integration testing are performed and documented in the Project File.
- Ensure that test documentation (e.g., test plans, test cases, and test procedures) are either updated or created.
- Identify, document, and resolve any risks exposed during software and test readiness reviews.
- Verify that the new product is placed under configuration management control.
- Verify that the training and technical documentation have been updated
- Verify the traceability of the design to the code.

Work Products: The output of the Construction Stage includes the following.

- Updated system.
- Updated design documentation.
- Updated test documentation.
- Updated user documentation.
- Updated training material.
- Statement of risk and impact to users.
- Test Readiness Review report.

Review Process: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.4-1. Construction Stage



Stage: **10.5 System Test Stage**

Responsibility: Independent Tester(s) or Project Team

Description: System testing is performed on the modified system. Regression testing is a part of system testing and is performed to validate that the modified code does not introduce faults that did not exist prior to the maintenance activity. *Exhibit 10.5-1* (provided at the end of the section) summarizes the input, process, control, and output for the System Test Stage.

Input: Input to the System Test Stage of maintenance includes the following.

- Test Readiness Review report.
- Documentation, which includes:
  - System test plans(s)
  - System test cases
  - System test procedures
  - User manuals
  - Design
- Updated system.
- Updated Project Plan.

Process: System tests are conducted on a fully integrated system. Testing shall include the performance of the following.

- System functional test.
- Interface testing.
- Regression testing.
- Test readiness review to assess preparedness for acceptance testing.

Results of tests conducted prior to the test readiness review should not be used as part of the system test report to substantiate requirements at the system level. This is necessary to assure that the test organization does not consider that testing all parts (one at a time) of the system constitutes a "system test."

Control: For maximum results, system tests should be conducted by an independent party. Prior to the completion of system testing, the test function is responsible for reporting the status of the activities that had been established in the test plan for satisfactory completion of system testing. The status is reported to the appropriate reviewers prior to proceeding to acceptance testing.

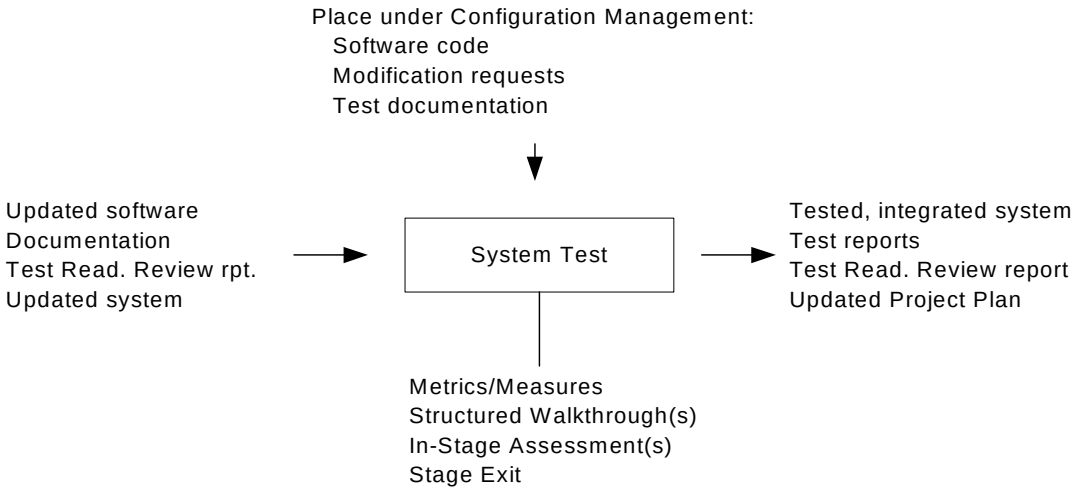
Code listings, Modification Requests, and test documentation are placed under configuration management. The system owner should participate in the review to ascertain that the maintenance release is ready to begin acceptance testing.

Work Products: The output for the System Test Stage of maintenance includes the following.

- Tested and fully integrated system.
- Test report.
- Test Readiness Review report.
- Updated Project Plan.

Review Process: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.5-1. System Test Stage



Stage: **10.6 Acceptance Stage**

Responsibility: System Owner/User or other designated individuals.

Description: Acceptance tests are conducted on a fully integrated system. Acceptance tests are performed by either the system owner, the user of the modified package, or a third party designated by the system owner. An acceptance test is conducted on the modified system, with the product that is under configuration management in accordance with the application's Configuration Management Plan. *Exhibit 10.6-1* (provided at the end of this section) summarizes the input, process, control, and output for the Acceptance Stage.

Input: Input for the Acceptance Stage of maintenance includes the following.

- Test Readiness Review report.
- Fully integrated system.
- Acceptance Test Plan.
- Acceptance test cases.
- Acceptance test procedures.

Process: The following steps form the process for acceptance testing.

- Perform acceptance tests at the functional level.
- Perform interoperability testing (to validate the functionality of any input and output interfaces).
- Perform regression testing.
- Conduct a Functional Configuration Audit (FCA).

The purpose of a FCA is to verify that all requirements specified and agreed to have been met. The FCA compares the system's elements (programs/modules) to the requirements documented in the current version of the Requirements Specification to assure that the modification addresses all, and only, those requirements. The results of the FCA should be documented, identifying all discrepancies found, and the plans for their resolution.

Control: Control of acceptance tests includes the following.

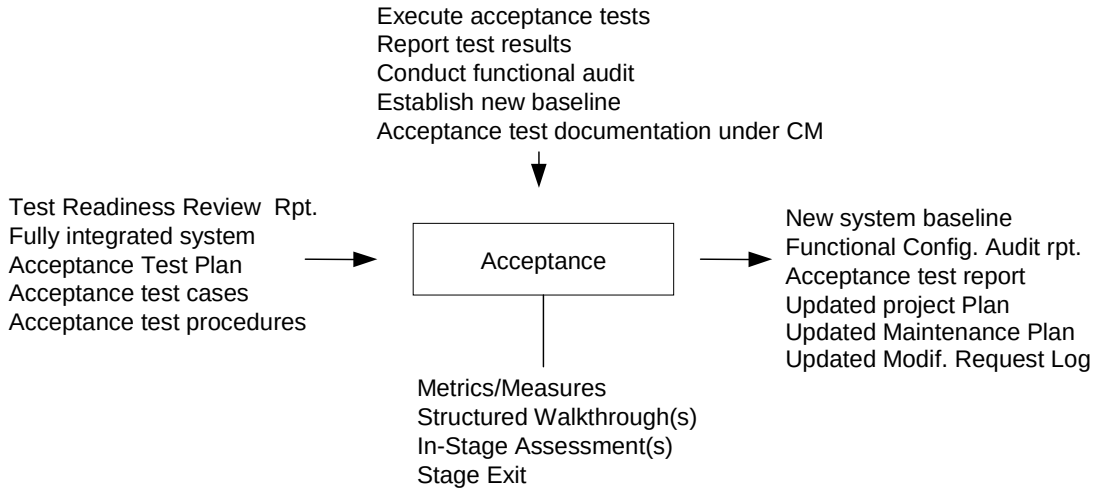
- Execute acceptance tests.
- Report results for the Functional Configuration Audit conducted to ensure that all of the functionality that has been agreed to is in fact present in the system.
- Receive approval from the change authority that the change has been successfully completed.
- Establish the new system baseline.
- Place the acceptance test documentation under CM.

Work Products: The output of the Acceptance Stage includes the following.

- New system baseline.
- Functional Configuration Audit Report.
- Acceptance Test Report.
- Updated Project Plan.
- Updated modification request log.
- Updated Maintenance Plan, as required for major enhancements.

Review Processes: Conduct structured walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.6-1. Acceptance Stage



Stage: **10.7 Delivery Stage**

Responsibility: Project Team

Description: This stage describes the requirements for the delivery of a modified system. *Exhibit 10.7-1* (provided at the end of the section) summarizes the input, process, control, and output for the Delivery Stage.

Input: Input to the Delivery Stage of maintenance is the fully tested version of the system as represented in the new baseline.

Process: The tasks for delivery of a modified system include the following:

- Conduct a Physical Configuration Audit (PCA).
- Notify the user community.
- Develop an archival version of the system for backup.
- Perform installation and training at the user facility.

The purpose of a PCA is to verify that the product associated with the modification and its documentation are internally consistent and are ready for delivery. The PCA compares the components (programs/modules) with its supporting documentation to assure that the documentation to be delivered correctly describes the system components. All discrepancies noted during the PCA, along with plans for their resolution, should be documented.

Control: Control for the Delivery Stage includes the following.

- Arrange and document a Physical Configuration Audit.
- Provide access to system materials for users, including replication and distribution.
- Complete the version description document.
- Place under configuration management control.

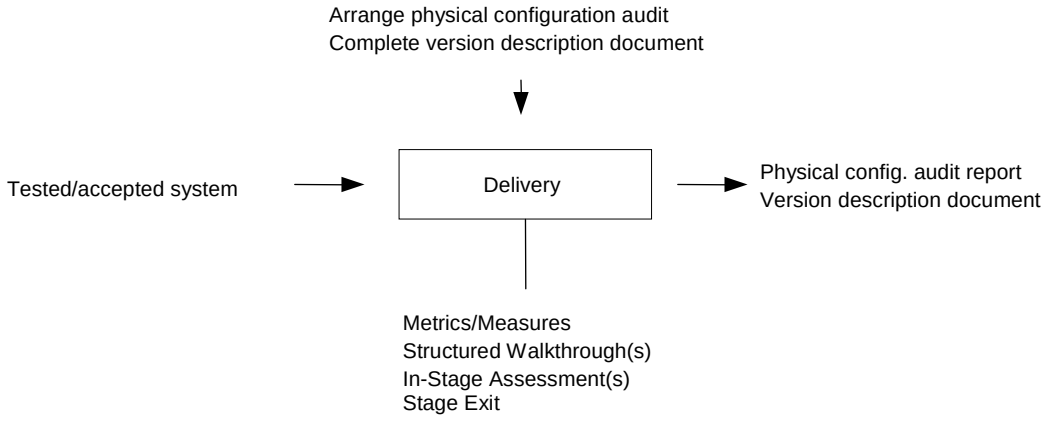
Work Products: The output of the Delivery Stage includes the following.

- Physical Configuration Audit report.
- Version Description Document (VDD).

The VDD contains information pertinent to the version or release of the system that is being delivered. Information provided includes system name, date delivered, version number, release number, brief description of functionality delivered in the modification, and prerequisite hardware and software with its associated version and release number. The current VDD is placed together with VDDs from previous versions/releases to form a complete chronology of the system from its initial implementation or Version 1, Release 1.

Review Process: Conduct Structured Walkthrough(s), In-Stage Assessment(s), and a Stage Exit.

Exhibit 10.7-1. Delivery Stage



Appendix A - Glossary

Acceptance criteria
The criteria that a software component, product, or system must satisfy in order to be accepted by the system owner or other authorized acceptance authority.

Acceptance process
The process used to verify that a new or modified system is fully operational and meets the system owner's requirements. Successful completion of the acceptance process results in the formal transfer of the system responsibilities from development to maintenance personnel.

Acceptance testing
Formal testing conducted to determine whether or not a software product or system satisfies its acceptance criteria and to enable the system owner to determine whether or not to accept the product or system.

Activity
A major unit of work to be completed in achieving the objectives of a project. An activity incorporates a set of tasks to be completed, consumes resources, and results in work products. An activity may contain other activities in a hierarchical manner. All project activities should be described in the Project Plan.

Algorithm
A finite set of well-defined rules for the solution to a problem in a finite number of steps. Any sequence of operations for performing a specific task.

Allocated requirements
The subset of the system requirements that are to be implemented within the scope of a given project, and forming the components of the system.

Anomaly
Anything observed in the operation or documentation of software and systems that deviates from expectations based on previously verified system or software products, or documents.

Application
Software or systems products designed to fulfill specific needs.

Assumption
A condition that is taken to be true without proof or demonstration.

Audit
An independent examination of a work product to assess compliance with specifications, standards, quality or security requirements, contractual agreements, or other predetermined criteria.

Baseline
A set of configuration items (hardware, software, documents) that has been formally reviewed and agreed upon, that serves as the basis for further development, and that can be changed only through formal change control procedures.

Baselined requirements
The set of project requirements that have been approved and signed off by the system owner during the Requirements Definition Stage. The system design is based on these requirements. The baselined requirements are placed under configuration control.

Code
Computer instructions and data definitions expressed in a development language or in a form that is output by an assembler, compiler, or other translator.

Code generator
A software tool that accepts as input the requirements or design for a computer program and produces source code that implements the requirements or design.

Code review
A meeting at which software code is presented to project personnel, managers, users, or other functional areas for review, comment, or approval.

Component
One of the parts that make up a system. A component may be hardware, software, or firmware and may be subdivided into other components.

Computer-Aided Software Engineering (CASE)
The use of computers to aid in the systems engineering process. May include the application of software tools for software design, requirements tracing, code production, testing, document generation, and other systems engineering activities.

Configuration control
An element of configuration management consisting of the evaluation, coordination, approval/disapproval, and implementation of changes to configuration items after formal establishment of their configuration identification.

Configuration Control Board (CCB)
A group of people responsible for evaluating and approving/disapproving proposed changes to configuration items, and for ensuring implementation of approved changes.

Configuration item
An aggregate of hardware, software, or documentation components that are designated for configuration management and treated as a single entity in the configuration management process.

Configuration Management
(1) A discipline that effectively controls and manages all modifications to system components, product, or system. Technical and administrative processes and tools are used to identify and document the functional and physical characteristics of the configuration items, manage and track changes to those items, record and report change processing and implementation status, and verify compliance with specified requirements.
(2) A Software Engineering Institute Capability Maturity Model key process area designed to establish and maintain the integrity of the software work products throughout the project's lifecycle.

Constraint
A restriction, limit, or regulation that limits a given course of action or inaction.

Construction Stage
The period of time in the project/product lifecycle during which a product is created from the design specifications. Testing is performed on the individual software units/components that have been coded, or on the combination of coded and purchased components, (e.g., as a COTS package.)

Cost estimate
A formal estimate of the cost to develop and support a project. Estimates should reflect all activities such as design, development, coding, testing, distribution, service, support of the product, staffing, training and travel expenses, subcontractor activities, contingencies,; and cost for external services (e.g., technical documentation production and Quality Assurance audits and reviews).

Deliverable
A work product that is identified in the Project Plan and is formally delivered to the system owner and other project stakeholders for review and approval.

Dependency
A relationship of one task to another where the start or end date of the second task is related to the start or end date of the first task.

Design
The process of defining the architecture, components, interfaces, and other characteristics of a system, product, or component.

Design specification
A document that describes the design of a software component, product, or system. Typical contents include architecture, control logic, data structures, input/output formats, interface descriptions, and algorithms.

Feasibility
The degree to which the requirements, design, or plans for a software product or system can be implemented under existing constraints.

Functional area
Any formally organized group involved in the development and maintenance of systems or the support of development and maintenance efforts, or other group whose input is required to successfully implement a systems project. Examples of functional areas include systems engineering services, technical writing, quality assurance, security, and telecommunications.

Functional Design Stage
The period of time in the project lifecycle during which the designs for architecture, software components, interfaces, and data are created, documented, and verified to satisfy project requirements.

Functional requirement
A requirement that specifies a function that a software component, product, or system must be able to perform.

Functional Test Plan
A plan for testing each function across one or more units. The plan describes how the functional testing occurs and the test procedure/test cases that will be used. The plan includes procedures for creating the test environment that allows all functions to be executed, the entry and exit criteria for starting and ending the function test period, and the schedule followed for starting and ending each test.

Functional test procedures
Procedures for each function or combination of functions to be tested. Procedures fully describe how the function is tested. Expected output from each test procedure is identified to compare the planned output to actual output.

Functional testing
Testing conducted to evaluate the compliance of a software product with specified functional requirements. Testing that focuses on the outputs generated in response to selected inputs and execution conditions.

Hardware
Physical computer and other equipment used to process, store, or transmit computer programs or data.

Hierarchy
A structure in which components are ranked into levels of subordination.

Implementation requirements
A requirement that supports the development and maintenance concepts and approaches in the areas of operating environment, conversion, installation, training, and documentation.

Incremental development
A development technique in which requirements definition, design, implementation, and testing occur in an overlapping, iterative (rather than sequential) manner, resulting in incremental completion of the overall system or product.

Information Engineering
A development methodology where models are created to improve the users' ability to understand and define the functions and flow of information within their organization. A business model is developed to identify the key areas of interest for the business, the tasks required for each area, and the activities that make up each task. The business model prioritizes and identifies top management goals and then establishes the information needs necessary to reach those goals. A data model is developed to describe the data and the relationships among data. The data model further divides the business model into user-defined relationships (e.g., entity relationship model).

Inspection
A static analysis technique that relies on visual examination of development products to detect errors, violations of development standards, and other problems. Code inspection and design inspection are two types.

Integration testing
An orderly progression of testing in which software components are combined and tested to evaluate the interaction between them.

Integrity
The degree to which a software component, product, or system prevents unauthorized access to, or modification of, computer programs or data.

Interactive analysis and design
A development methodology that uses facilitated team techniques, such as Joint Application Development or Rapid Application Development, to rapidly develop project requirements that reflect the users' needs in terminology that the users understand. Group facilitation techniques are especially important when several user organizations have unique project requirements that are specific to their mission and goals.

Interface requirement
A requirement that specifies an external item with which a software product or system must interact, or that sets forth constraints on formats, timing, or other factors caused by such an interaction.

Interface testing
Testing conducted to evaluate whether system components pass data and control correctly to one another.

Interview technique
A technique for the identification, analysis, and documentation of the project requirements. The project team conducts a series of interviews with users to identify the users' perceived IT functional needs, analyzes the information gathered during the interviews, and develops the requirements.

Key Process Area
Software engineering processes identified by the Software Engineering Institute Capability Maturity Model where a project team should focus its efforts to achieve consistently high quality software products.

Lifecycle
See Project Lifecycle.

Maintenance
The process of supporting a software product or system after delivery to maintain operational status, correct faults, improve performance or other attributes, or adapt to a changed environment.

Menu-driven
Pertaining to a system or mode of operation in which the users direct the software through menu selections.

Methodology
A collection of methods, procedures, and standards that defines an integrated synthesis of engineering approaches to the development of a work product.

Milestone
A scheduled event for which an individual or team is accountable and that is used to measure progress.

Module
A program unit that is discrete and identifiable with respect to compiling, combining with other units, and loading. A logically separable part of a program.

Module testing
Testing of individual software modules or groups of related modules to verify the implementation of the design.

Organization
An organization is a unit within a company or other entity within which projects are managed as a whole. Examples of an organization include, the Department of Energy, a contractor organization, a program (e.g., RW, or a laboratory.) All projects within an organization share a common top-level manager and common policies.

Performance requirement
A requirement that imposes conditions on a functional requirement (e.g., a requirement that specifies the speed, accuracy, or memory usage with which a given function must be performed).

Planning Stage
The initial stage in the project lifecycle during which the system owner/users' needs and expectations are identified, the feasibility of the project is determined, and the Project Plan is developed.

Platform
A specific computer and operating system on which a software product or system is developed or operated.

Portability
The ease with which a software component, product, or system can be transferred from one hardware or software environment to another.

Procedure
A written description of a course of action to be taken to perform a given task.

Process
An ordered set of steps performed for a given purpose. Processes define or control the development of the project work products. The use of processes will ensure a consistent methodology across all platforms in producing the lifecycle deliverables.

Product
See Work product.

Developer's Reference Manual
A work product deliverable that provides information necessary to maintain or modify components for a given computer system. Typically described are the equipment configuration, operational characteristics, coding features, input/output features, and compilation or assembly features of the computer system.

Project
An undertaking requiring concerted effort that is focused on developing or maintaining a specific software product or system. A project has a distinct beginning and end, and has its own funding, cost accounting, and delivery schedule.

Project file
A central repository of material and artifacts pertinent to a project. Contents typically include all work products, memos, plans, technical reports, and related items.

Project lifecycle
Covers all activities conducted within the scope of an entire project, from project startup to project closeout.

Project Management Plan
See Project Plan.
Synonymous with software development plan and project plan.

Project Manager
The individual with total responsibility for all activities of a project. The project manager plans, directs, controls, administers, and regulates a project.

Project Plan
A document that describes the technical and management approach to be followed for a project. The plan typically describes the work to be done, the resources required, the methods to be used, the procedures to be followed, the schedules to be met, and the way the project will be organized.

Project planning
A Software Engineering Institute Capability Maturity Model key process area designed to establish reasonable plans for performing systems engineering and for managing the project.

Project team
The project manager, analysts, developers, and other staff assigned as the core group for a project. The project team may include representatives of the other functional areas (e.g., technical writer and telecommunications expert) responsible for contributing to the development, installation, and maintenance of the software product.

Project Test Plan
Defines the what, how, where, and when about all test activities required to assure that the software product or system will perform satisfactorily for all users. As a minimum, the plan should include descriptions for unit testing, integration testing, system testing, and acceptance testing.

Project tracking and oversight
A Software Engineering Institute Capability Maturity Model key process area designed to provide adequate visibility into actual project progress so that management can take effective actions when the project's performance deviates significantly from the plans.

Prototyping
A technique for developing and testing a preliminary version of the software product (either as a whole or in modular units) in order to emulate functionality without such encumbering features as error handling, help messages, security controls, and other utilities that are not part of the design logic. This allows the project team to test the overall logic and workability of required functions and provides a model by which the project team and users can jointly determine if the software requirements meet the intended objectives. Prototyping is often used in conjunction with interactive analysis and design techniques.

Pseudocode
A combination of development language constructs and natural language used to express a computer program design.

Quality assurance
A process designed to provide management with appropriate visibility into the systems engineering processes being used by the project team and the work products being built. One of the Software Engineering Institute Capability Maturity Model level 2 key process areas.

Rapid Prototyping
A type of prototyping in which emphasis is placed on developing prototypes earlier in the development process to permit early feedback and analysis in support of the development process.

Reference
A document(s) or other material that is useful in understanding more about an activity.

Regression testing
Selective retesting of a software or system component to verify that modifications have not caused unintended effects and that the software or system component still complies with its specified requirements.

Reliability
The ability of a software or system component to perform its required functions under stated conditions for a specified period of time.

Requirement
A condition or capability needed by a system owner/user to solve a problem or achieve an objective. A condition or capability that must be met or possessed by the software product or system to satisfy a contract, standard, specification, or other formally imposed documents.

Requirements analysis
The process of analyzing and understanding the scope and feasibility of identified requirements; of developing a preliminary plan to arrive at a detailed definition of system, hardware, or software requirements; and of crystallizing a preliminary system solution.

Requirements Definition Stage
The period of time in the project lifecycle during which the requirements for an IT product are defined and documented.

Requirements management
A process designed to establish a common understanding between the system owner/users and the project team regarding the system owner/users' software and system requirements. This understanding forms the basis for estimating, planning, performing, and tracking the project's activities throughout the lifecycle. One of the Software Engineering Institute Capability Maturity Model level 2 key process area

Requirements Specification
A work product deliverable that specifies the manual and automated requirements for a software product or system in non-technical language that the system owner/users can understand. Typically included are functional requirements, performance requirements, and interface requirements. Describes in detail what will be delivered in the product or system release.

Retirement
Permanent removal of a system or software product from its operational environment.

Reusability
The degree to which a software module or other work product or system component can be used in more than one computer program or software system.

Reverse Engineering
A development methodology in which the software development process is performed in reverse. The technique involves the examination of an existing software product that has characteristics that are similar to the desired product. Using the existing code as a guide, the requirements for the product are

defined, analyzed, and abstracted all the way back to specifications. Any required code changes can be made based on a specification-like format. Ideally, the specifications would be edited and passed to a code generator that would trigger automatic documentation and revisions. Once testing is complete, the revised code is placed into production.

Risk
The possibility of suffering loss.

Risk management
An approach to problem analysis that is used to identify, analyze, prioritize, and control risks.

Software
Computer programs, procedures, and associated documentation and data pertaining to the operation of a software product or system.

Software Development Plan
DOD usage, see Project Plan.

Software Quality Assurance
See Quality Assurance.

Systems Engineering Methodology
The Departmental methodology that identifies the processes, activities, tasks, management responsibilities, and work products that are required for each system development and maintenance project. Deviations from the methodology require the approval of all stakeholders who have approval rights on the project. A key objective of the methodology is to provide measurable, repeatable processes to assure that project development and maintenance methodologies are consistent throughout the Departmental information technology environment.

System
A product and the documentation, hardware, and communications needed to implement and operate the product and accomplish a specific function or set of functions.

Specification
A document that specifies in a complete, precise, verifiable manner the requirements, design, behavior, and other characteristics of a software component, product, or system.

Spiral development model
A software development process in which the constituent activities, typically requirements analysis, design, coding, integration, and testing are performed iteratively until the software product is complete.

Stage
A partition of the project lifecycle that reduces a project to manageable pieces and represents a meaningful and measurable set of related tasks that are performed to obtain specific work products.

Stakeholder
The DOE individual(s) with decision-making authority over a project or group of projects.

Standard
Mandatory requirements employed and enforced to prescribe a disciplined, uniform approach to software and systems development and maintenance.

Structured analysis
An analysis technique that uses a graphical language to build models of software products or systems. The four basic features in structured analysis are data flow diagrams, data dictionaries, procedure logic representations, and data store structuring techniques.

System
A collection of hardware, software, firmware, and documentation components organized to accomplish a specific function or set of functions.

System Design Document
A work product deliverable that describes the solution to the automation task as described by the requirements. Contains sufficient detail to provide necessary direction for writing the Program Specifications and allows developers maximum technical freedom.

System Design Stage
A stage in the lifecycle model during which the designs for the software product or system architecture, software components, interfaces, and data are refined and expanded to the extent that the design is sufficiently complete to be implemented.

Systems engineering
An inter-disciplinary approach and means to enable the realization of successful systems.

System owner
The organizational unit that funds and has approval authority for the project. Typically, system owners are also system users.

System testing
Testing conducted on a complete, integrated software product or system to evaluate compliance with its specified requirements.

Task
The smallest unit of work subject to management accountability. A task is a well-defined work assignment for one or more project team members. Related tasks are usually grouped to form activities. A task is the lowest level of work division typically included in the Project Plan and Work Breakdown Structure.

Telecommunications
The science and technology of communications by electronic transmission of impulses, as by telephone or e-mail.

Test bed
An environment containing the hardware, instrumentation, simulators, software tools, and other support elements needed to conduct a test.

Test case
A set of test inputs, execution conditions, and expected results that are developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement.

Test criteria
The criteria that a software product or system component must meet in order to pass a given test.

Test design
Documentation specifying the details of the test approach for a software or system feature or combination of features and identifying the associated tests.

Test documentation
Documentation describing plans for, or results of, the testing of a system component or product. Documents typically include test case specifications, test incident reports, test logs, test plans, test procedures, and test reports.

Test item
A system component that is the object of testing.

Test log
A chronological record of all relevant details about the execution and results of a test.

Test phase
The period of time in the project lifecycle in which the components of a system are evaluated and integrated, and the product is evaluated to determine whether or not the requirements have been satisfied.

Test plan
A document specifying the scope, approach, resources, and schedule of intended testing activities. The plan identifies test items, the features to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning.

Test procedure
Detailed instructions for the setup, execution, and evaluation of the results for a given test case.

Test report
A document that describes the conduct and results of the testing carried out for a software or system component or product.

Testing
An activity in which a software or system component or product is executed under specified conditions, the results are observed and recorded, and an evaluation is made.

Traceability
The degree to which a relationship can be established between two or more products of the development process, especially products having a predecessor-successor relationship to one another.

Transaction analysis
A technique used to derive structured charts for a software product that will process transactions. Transaction analysis is used to divide complex data flow diagrams into smaller, simpler data flow diagrams--one for each transaction that the product or system will process. Structure charts are developed from the simple data flow diagrams. The individual structure charts for the separate transactions are then combined to form one large structure chart that is very flexible and can accommodate user changes.

Unit
A separately testable element specified in the design of a computer system or software component. A software or system component that is not subdivided into other components.

Unit testing
Testing of individual hardware or software units or groups of related units. The isolated testing of each flowpath of code with each unit. The expected output from the execution of the flowpath should be identified to allow comparisons of the planned output against the actual output.

Usability
The ease with which a user can learn to operate, prepare inputs for, and interpret outputs of an IT product or system.

User
Within the context of information systems, the general population of individuals who use a software product or system. User activities can include data entry; read only; add, change and delete capabilities; querying; and report generation.

User interface
An interface that enables information to be passed between a user and hardware or software components of a computer system.

User manual
A document that presents the information necessary to use a software product or system to obtain desired results. Typically described are product or component capabilities, limitations, options, permitted inputs, expected outputs, possible error messages, and special instructions.

Validation
The process of evaluating software or systems at the end of the development process to assure compliance with established software and system requirements.

Verification
The process of evaluating a software product or system to determine whether or not the work products of a stage of the project lifecycle fulfill the requirements established during the previous stage.

Walkthrough
An analysis technique in which a team of subject matter experts review a segment of code, documentation, or other work product, ask questions, and make comments about possible errors, violation of development standards, and other problems.

Work product
Any tangible item that results from a project function, activity, or task. Examples of work products include process descriptions, plans, procedures, computer programs, and associated documentation, which may or may not be intended for delivery to the system owner and other project stakeholders.

Bibliography: The following materials were used in the preparation of the glossary.

1. Carnegie Mellon University, Software Engineering Institute, *Capability Maturity Model: Guidelines for Improving the Software Process*, Addison Wesley Longman, Inc., 1994.

2. Software Engineering Institute, Software Process Maturity Questionnaire, Capability Maturity Model, version 1.1, Carnegie Mellon University, Pittsburgh, 1994.

3. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard Glossary of Software Engineering Terminology*, IEEE Std 610.12-1990, The Institute of Electrical and Electronics Engineers, Inc., New York, 1990.

4. Webster's II New Riverside University Dictionary, Houghton Mifflin Company, Boston, 1984.

Appendix B – List of Abbreviations

ABC	Analysis of Benefits and Costs
ACPPM	Assistant Computer Protection Program Manager
ANSI	American National Standards Institute
CASE	Computer-Aided Software Engineering
CIO	Chief Information Officer
CM	Configuration Management
CMM	Capability Maturity Model
COTS	Customized Off-The-Shelf
CPIC	Capital Planning and Investment Control
CPP	Computer Protection Plan
CSSO	Computer Systems Security Officer
DOE	Department of Energy
DOS	Disk operating system
EA	Enterprise Architecture
ESTSC	Energy Science and Technology Software Center
FCA	Functional Configuration Audit
IBM	International Business Machines
IEEE	The Institute of Electrical and Electronics Engineers, Inc.
IP	Internet Protocol
IRM	Information Resource Management
ISA	In-Stage Assessment
IT	Information Technology
LAN	Local area network
PCA	Physical Configuration Audit
POC	Point of contact
SEI	Software Engineering Institute (at Carnegie-Mellon)
SEM	Systems Engineering Methodology
Std	Standard
SWT	Structured Walkthrough
TCP	Transmission Control Protocol
VDD	Version Description Document
WAN	Wide area network

Appendix C - Index