Software Engineering Methodology

Chapter 7.0
Programming Stage

Table of Contents

Chapter **Page**

7.0 Programming Stage 7.0-1
- 7.1 Develop Acquisition Plan 7.1-1
- 7.2 Develop Installation Plan 7.2-1
- 7.3 Establish Programming Environment 7.3.1
- 7.4 Write Programs 7.4-1
- 7.5 Conduct Unit Testing 7.5-1
- 7.6 Establish Development Baselines 7.6-1
- 7.7 Plan Transition to Operational Status 7.7-1
- 7.8 Generate Operating Documentation 7.8-1
  - 7.8.1 Produce Users Manual 7.8-3
  - 7.8.2 Produce Programmers Reference Manual 7.8-5
- 7.9 Develop Training Program 7.9-1
- 7.10 Revise Project Plan 7.10-1
- 7.11 Conduct In-Stage Assessment 7.11-1
- 7.12 Conduct Programming Stage Exit 7.12-1

Chapter: **7.0**
Programming Stage

Description: In this stage any hardware or software procured to support the programming 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 program is produced.

The activities in this stage result in the transformation of the system design into the first complete representation of the software product. The source code, including suitable comments, is generated using the approved program specifications. If the software product requires a data base, any data base utilities are coded. 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 software 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
- Programming Standards
- Project Plan *(revised)*
- Software 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 software 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 software 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 Programming Environment
7.4 Write 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
7.10 Revise Project Plan
7.11 Conduct In-Stage Assessment
7.12 Conduct Programming Stage Exit

Output:

Several work products are produced during this stage. The work products listed below are the minimum requirements for a large software 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)*
- Software 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
  - Programmers Reference Manual
- Training Plan *(draft)*
- Project Plan *(revised)*

Output, continued: A matrix showing the work products associated with each high-level activity is provided in *Exhibit 7.0-1, Programming 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: Structured walkthroughs are necessary during this stage to validate work products. The activities that are appropriate for structured walkthroughs are identified throughout the chapter. The time and resources needed to conduct the walkthroughs should be indicated in the project resources, schedule, and work breakdown structure.

References: *Appendix C, Conducting Structured Walkthroughs,* provides a procedure and sample forms that can be used for structured walkthroughs.

Exhibit 7.0-1. Programming Stage Activities and Work Products by Project Size

Work Activity		Project Size			Work Product	Scheduled Deliverables		
		L	M	S		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 Programming Environment	R	R	R		N	N	N
7.4	Write Programs	R	R	R	Completed units and modules of code	I	I	I
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 R	I R R R	I R R R
7.6	Establish Development Baselines	R	R	R	Baselined code	I	I	I
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)* Programmers Reference Manual *(draft)*	R R	R R	R R
7.9	Develop Training Program	R	R	A	Training Plan *(draft)*	R	R	A
7.1	Revise Project Plan	R	R	A	Project Plan *(revised)*	R	R	A
7.11	Conduct In-Stage Assessment	R	R	A	ISA Report Form[1]	N	N	N
7.12	Conduct Programming 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 Applicable

I = Input to other deliverables
[1] = Completed by reviewer
[2] = Can use existing plan/procedure

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

1. *Software Engineering Handbook*, Chapter 7, Software Testing.

2. The Institute of Electrical and Electronics Engineers, Inc., *IEEE Standard for Developing Software Life Cycle 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 1063-1987, New York, 1988.

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 Life Cycle Standards Manual,* 1991.

7. U.S. Department of Labor, Directorate of Information Resources Management, *Systems Engineering Concepts and Procedures Manual,* 1988.

8. U.S. Department of Labor, Directorate of Information Resources Management, *Systems Engineering Standards Manual,* 1988.

Activity: **7.1**
Develop Acquisition Plan

Responsibility: Project Team

Description: Develop a plan for the acquisition of any hardware, software, and telecommunications equipment needed to install and operate the software product at all system owner and user sites. The plan should address any special procurements necessary to accommodate the hardware and tele-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 telecommunications 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.

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 software 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 telecommunications 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 telecommunications 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.
- Telecommunications requirements.
- Dependencies among activities affected by installation.
- Installation tests to assure the integrity and quality of the installed product.

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 Software Integration and Testing Stage.

Activity: **7.3**
Establish Programming Environment

Responsibility: Project Team

Description: Establishing the programming environment involves assembling and installing the hardware, software, telecommunications equipment, data bases, and other items required to support the programming 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 programming environment, it may be necessary to alter the operational environment to accommodate an infrastructure of purchased hardware and software for use during programming and testing.

Before being integrated into, or used to support, the software 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 programming standards 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**
Write Programs

Responsibility: Project Team Programmers

Description: This activity involves generating the source and object code for the software product. The code should be written in accordance with the programming standards developed in the System Design Stage. Regardless of the platform, development of the code should adhere to a consistent set of programming techniques and error prevention procedures. This will promote reliable, maintainable code, developed in the most efficient and cost effective manner.

The source and object code should be uniquely identified and stored in a way to facilitate the configuration control measures described in the Software Configuration Management Plan.

Writing 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 and machine-readable modules.
- Generate the physical files and data base structure.
- Generate video screens, report generation codes, and plotting instructions.
- 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 programming staff should meet at scheduled intervals to discuss problems encountered and to facilitate program integration and uniformity.

Coding Practices, continued:

- 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.
- 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.
- Test materials generated from preliminary testing.

Review Process: Weekly informal reviews of each programmer'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 software product and support the early detection of errors in design, logic, or code.

Conduct structured walkthroughs on 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, conduct code inspections at successive stages of code production. These inspections are particularly important when code is being produced by several programmers or different programming 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 Programmers

Description: Unit testing is used to verify the input and output for each module. Successful testing indicates the validity of the function or subfunction 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 software 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 programmer, the project manager or lead programmer should be aware of the unit test results.

Work Products: Completion of unit testing for a software 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 Software Configuration Management Plan. Configuration controls restrict changes to tested and approved software.

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 software 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.

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.

Activity: **7.6**
Establish Development Baselines

Responsibility: Project Team Programmers

Description: A development baseline is an approved "build" of the software product. A build can be a single component or a combination of software components. The first development baseline is established after the first build is completed, tested, and approved by the project manager or lead programmer. 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, and performance/ 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 Software Configuration Management Plan. Approved changes to a development baseline must be incorporated into the next build of the software product and revisions made to the affected work products (e.g., Software 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 Software 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.

Activity: **7.7**
Plan Transition to Operational Status

Responsibility: Project Team

Description: Successful transition from acceptance testing to full operational use of the software product depends on planning the transition long before the software product is installed in its operational environment. In planning for the transition, quantify the operational needs associated with the software 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 software product.
- Define the configuration management procedures that will be used for the operational software product. Ideally, the methods defined in the Software 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 software 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.

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 software product from the users point-of-view. The different ways that users (including system administration and maintenance personnel) will interact with the software 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 software project.

- Users Manual
- Programmers Reference Manual
- Systems Administration Manual
- Data Base 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 software 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 software 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 software 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 software 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 software product goes into production. Operating documents must be kept up-to-date as long as the software product remains in production.

Tasks: The following tasks describe the minimum requirements for operating documentation.

7.8.1 Produce Users Manual
7.8.2 Produce Programmers Reference Manual

Task: **7.8.1**
Produce Users Manual

Responsibility: Project Team/Technical Writer

Description: The Users Manual provides detailed information users need to access, navigate through, and operate the software product. Users rely on the Users Manual to learn about the software 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 software 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 data base 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, nontechnical 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 software product.
- Instructions for how to install, setup, or access the software product.
- Complete coverage of all software 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 software product that are most difficult to understand.
- Clear delineation of which features are accessible only to specific users.

Work Product,

continued:

- Instructions on accessing and using online help features.
- Procedures for data entry.
- 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 software products, separate manuals (e.g., User's Manual, Data Base 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 software product and its functions.

The draft user documentation will be tested and verified with the software product during the Software Integration and Testing Stage.

Task: **7.8.2**
Produce Programmers Reference Manual

Responsibility: Project Team/Technical Writer

Description: The Programmers Reference Manual contains programming information used by the maintenance staff to maintain the programs, data bases, interfaces, and operating environment. The Programmers Reference Manual should provide an overall conceptual understanding of how the software product is constructed and the details necessary to implement corrections, changes, or enhancements.

The Programmers Reference Manual describes the logic used in developing the software product and the functional and system flow to help the maintenance programmers understand how the programs fit together. The information should enable a programmer 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 Programmers Reference Manual.

- A description of the technical environment, including versions of the programming 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 software architecture, program structure, and program calling hierarchy.
- The design and programming standards and techniques used to develop the software product.
- Concise descriptions of the purpose and approach used for each program.
- Layouts for all data structures and files used in the software 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.

Work Product, continued:

- Manual and automated backup procedures.
- Error processing features.

Use appendixes to provide detailed information that is likely to change as the software 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 Programmers Reference Manual to assure that the documentation is complete, easy to use, and accurately reflects the software product and its functions.

The draft Programmers Reference Manual will be tested and verified with the software product during the Software 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 software 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 equipment 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 software 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 software 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 software 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).

Work Product: Prepare a draft Training Plan that describes the Training Program and 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.

Work Product,

continued:

- 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 software product's installation and conversion to production status.
- 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 Software Integration and Testing Stage.

Review Process: Conduct a structured walkthrough to assure that the draft Training Plan is accurate and complete.

Activity: **7.10**
Revise Project Plan

Responsibility: Project Manager

Description: Once the coding effort is completed and unit and integration testing have been conducted, determine if the project estimates for resources, cost, and schedule need to be revised.

Work Product: Review the Project Plan for accuracy and completeness of all Programming Stage activities and make any changes needed to update the information. Expand the information for the Software Integration and Testing Stage to reflect accurate estimates of resources, costs, and hours. Place a copy of the revised Project Plan in the Project File.

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

Review Process: Conduct a structured walkthrough to ensure that the Project Plan reflects the project's current status and adequately estimates the resources, costs, and schedule for the Software Integration and Testing Stage.

The Project Plan is formally reviewed during the In-Stage Assessment and Stage Exit processes.

Activity: **7.11**
Conduct In-Stage Assessment

Responsibility: Project Manager and Independent Reviewer

Description: An In-Stage Assessment (ISA) is an independent review of the work products and deliverables developed or revised during each stage of the project lifecycle. The independent reviewer is typically a member of the Quality Assurance Team who is assigned to the software project and conducts all of the ISAs for the project.

An ISA does not require meetings with, or extra work by, the project team. All of the work products and deliverables needed for the review should be readily available in the Project File.

Schedule at least one ISA prior to the Programming Stage Exit process. Additional ISAs can be performed during the stage, as appropriate.

Provide the reviewer with copies of all work products developed or revised during the Programming Stage including the Project Plan. The reviewer assesses the work products and deliverables to verify the following:

- The project is complying with the site's software engineering standards/best practices.
- Sound project management practices are being used.
- The project risks are identified and mitigated.

A description of the ISA process and the ISA report form are provided in the *In-Stage Assessment Process Guide*. A copy of the guide is provided in Appendix D.

Note: An ISA is an effective project management tool that is recommended for all projects regardless of size.

Work Product: 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.

Activity: **7.12**
Conduct Programming Stage Exit

Responsibility: Project Manager

Description: The Stage Exit is a process for ensuring that projects are on target, within budget, on schedule, and meet the DOE and project standards identified in the Project Plan. The goal of a Stage Exit is to secure the approval of designated key individuals to continue with the project and to move forward into the next lifecycle stage.

Schedule the Stage Exit as the last activity of the Programming Stage. It is the responsibility of the project manager to notify the appropriate participants when a project is ready for the Stage Exit process and to schedule the Stage Exit meeting. All functional areas and the Quality Assurance representative involved with the project should receive copies of the work products and deliverables produced in this stage.

During the Stage Exit meeting, participants discuss open issues that will impact the Project Plan. The project manager should ensure that an acceptable action plan is developed for handling all open issues. At the conclusion of the meeting, concurrence is needed from the designated approvers to begin the next stage.

A description of the Stage Exit process is provided in the *Stage Exit Process Guide.* A copy of the guide is provided in Appendix E.

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

Work Product: 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 Software Integration and Testing Stage.