Software Engineering Methodology

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 software product is fully operational and meets the system owner's requirements. Successful completion of the acceptance process results in the formal transfer of the software product 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 software 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.

Anomaly
Anything observed in the operation or documentation of software that deviates from expectations based on previously verified software products or documents.

Application
Software 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 (software components and 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 software product design is based on these requirements. The baselined requirements are placed under configuration control.

Code
Computer instructions and data definitions expressed in a programming 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 software engineering process. May include the application of software tools for software design, requirements tracing, code production, testing, document generation, and other software 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
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 or software components that are designated for configuration management and treated as a single entity in the configuration management process.

Configuration management See Software configuration management

Constraint
A restriction, limit, or regulation that limits a given course of action or inaction.

Cost estimate
A formal estimate of the cost to develop and support a project. Estimates should reflect all activities such as design, development, coding, distribution, service, and 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 software 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 software or the support of development and maintenance efforts, or other group whose input is required to successfully implement a software project. Examples of functional areas include software engineering services, technical writing, quality assurance, security, and telecommunications.

Functional Design Stage
The period of time in the software 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 testing 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 software 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 software 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 software components pass data and control correctly to one another.

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 Software 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.

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 software 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 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.

Programmers Reference Manual
A work product deliverable that provides information necessary to maintain or modify software for a given computer system. Typically described are the equipment configuration, operational characteristics, programming features, input/output features, and compilation or assembly features of the computer system.

Programming Stage
The period of time in the software lifecycle during which a software product is created from the design specifications and testing is performed on the individual software units.

Project
An undertaking requiring concerted effort that is focused on developing or maintaining a specific software product or system. A project has its own funding, cost accounting, and delivery schedule.

Project File
A central repository of material pertinent to a project. Contents typically include all work products, memos, plans, technical reports, and related items.

Project lifecycle
The software lifecycle selected for the project and approved by the system owner and other project stakeholder(s).

Project manager
The individual with total business responsibility for all software activities of a project. The project manager 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. The plan includes a list of deliverables, actions required, and other key events needed to accomplish the project.

Project Team
The project manager, analysts, programmers, 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 all test activities required to assure that the software product will perform satisfactorily for all users. As a minimum, the plan should include descriptions for unit testing, integration testing, system testing, and acceptance testing.

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 programming language constructs and natural language used to express a computer program design.

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 component to verify that modifications have not caused unintended effects and that the software component still complies with its specified requirements.

Reliability
The ability of a software 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 to satisfy a contract, standard, specification, or other formally imposed documents.

Requirements analysis
The process of studying system owner/user(s) needs to arrive at a definition of system, hardware, or software requirements.

Requirements Definition Stage
The period of time in the software lifecycle during which the requirements for a software product are defined and documented.

Requirements management
A Software Engineering Institute Capability Maturity Model key process area designed to establish a common understanding between the system owner/user and the project team regarding the system owner/users' software requirements. This understanding forms the basis for estimating, planning, performing, and tracking the project's activities throughout the lifecycle.

Requirements Specification
A work product deliverable that specifies the manual and automated requirements for a software product in nontechnical 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 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 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 configuration item
An aggregation of software that is designated for configuration management and treated as a single entity in the configuration management process.

Software configuration management
(1) A discipline that effectively controls and manages all modifications to a software component, 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.

Software Engineering Methodology
The Departmental methodology that identifies the processes, activities, tasks, management responsibilities, and work products that are required for each software development and maintenance project. Deviations from the methodology require the approval of all parties 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 systems environment.

Software lifecycle
The period of time that begins when a software product is conceived and ends when the software is retired. A network of stages and processes that function together to guide the development and maintenance of software products. Each process produces a set of deliverables as it moves through the lifecycle.

Software Process Planning
A Software Engineering Institute Capability Maturity Model key process area designed to establish reasonable plans for performing software engineering and for managing the software project.

Software 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.

Software Quality Assurance
A Software Engineering Institute Capability Maturity Model key process area designed to provide management with appropriate visibility into the software engineering processes being used by the project team and the work products being built.

Software System
A software product and the documentation, hardware, and telecommunications 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, or 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 software lifecycle that reduces a project to manageable size 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 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 architecture, software components, interfaces, and data are refined and expanded to the extent that the design is sufficiently complete to be implemented.

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.

User
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.

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. An task is the lowest level of work division typically included in the Project Plan and Work Breakdown Structure.

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 component or product must meet in order to pass a given test.

Test design
Documentation specifying the details of the test approach for a software feature or combination of software features and identifying the associated tests.

Test documentation
Documentation describing plans for, or results of, the testing of a software component or product. Documents typically include test case specifications, test incident reports, test logs, test plans, test procedures, and test reports.

Test item
A software item 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 software lifecycle in which the components of a software product are evaluated and integrated, and the software 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 component or product.

Testing
An activity in which a software 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.

Interview technique
An 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 automated functional needs, analyzes the information gathered during the interviews, and develops the requirements.

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 software component. A software 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 a software product.

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 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 at the end of the software development process to assure compliance with established software and system requirements.

Verification
The process of evaluating a software product to determine whether or not the work products of a stage of the software 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 or documentation, 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. Software Engineering Institute, Software Process Maturity Questionnaire, Capability Maturity Model, version 1.1, Carnegie Mellon University, Pittsburgh, 1994.

2. 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.

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