

Appendices

Table of Contents

Appendix A: Systems Engineering Methodology Glossary - Key Terms and Acronyms

Appendix B: SEM Templates, Checklists and Guides Listing – In Order by Chapter / Phase

Appendix C: Investigate Alternatives

Appendix D: List of Acronyms

Appendix E: Roles and Responsibilities

Page inserted for consistency in section start points.

Appendix A – Systems Engineering Methodology Glossary

Key Terms and Acronyms

The following is a list of common terms and acronyms used within the Systems Engineering industry. While many of these terms are not mentioned within the body of this guide, they are nonetheless important to understanding this Systems Engineering Methodology (SEM). These terms are taken from several sources, including State of Michigan procedure documents, the State of Michigan Department of Information Technology Project Management Methodology, and the referenced sources contained in the SEM. If you need further information on any of the subjects in the following list, please consult the SEM, the variety of sources listed in the Websites listing (page viii) and the sources listed in the Resources and Reference lists contained throughout the SEM.

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 Integrated 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.

Terms used in Describing Deliverables:

Initial – the Stage indicated is the first time the deliverable is worked on

Final – all work on the deliverable has been completed during the Stage indicated

Revised – the deliverable was worked on during the Stage indicated

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.

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.

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.

Glueware

Integration software that provides the proper interface for the component (i.e., wrappers) being integrated and serves as a mediator for its interactions with other components.

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).

Initiation and 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 Charter and Project Plan are developed.

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 Integrated 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, MDIT, 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).

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.

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 Methodology

The State of Michigan document that provides governance for all State of Michigan projects.

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 Integrated 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 Integrated 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 Integrated 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 Integrated 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 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 Quality Assurance

See Quality Assurance.

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 State of Michigan 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

(1) A collection of hardware, software, firmware, and documentation components organized to accomplish a specific function or set of functions.

(2) A product and the documentation, hardware, and communications needed to implement and operate the product and 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.

Systems Engineering Methodology

The MDIT 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 MDIT information technology environment.

System owner

The enterprise 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 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.

Testing Stage

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.

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 – SEM Templates, Checklists and Guides**In Order by Chapter / Phase**

The following is a list of the SEM templates/checklists and SEM process guides referenced in the SEM. They are ordered as they appear in the chapters of this document.

Chapter 1:	None.
Chapter 2:	None.
Chapter 3:	Maintenance Plan (SEM-0301) Software Configuration Management Plan (SEM-0302)
Chapter 4:	Requirements Traceability Matrix (SEM-0401) Requirements Specification (SEM-0402) Requirements Management Checklist (SEM-0403)
Chapter 5:	Functional Design Document (SEM-0501)
Chapter 6:	Conversion Plan (SEM-0601) Test Plan (SEM-0602) Test Report (SEM-0603) System Design Document (SEM-0604) System Design Checklist (SEM-0605) Software Testing Checklist (SEM-0606)
Chapter 7:	Transition Plan (SEM-0701) Installation Plan (SEM-0702) Training Plan (SEM-0703) Training Checklist (SEM-0704)
Chapter 8:	Integration and System Testing Checklist (SEM-0801) Error Reporting and Tracking Checklist (SEM-0802) Pre-Acceptance Checklist (SEM-0803) Testing Package Checklist (SEM-0804) User Acceptance Checklist (SEM-0805)
Chapter 9:	None.

The Structured Walkthrough Process Guide and the Stage Exit Process Guide are utilized in all SEM stages.

Page inserted for consistency in section start points.

Appendix C – Investigate Alternatives

Activity:	C-1. Investigate Alternatives
Responsibility:	Project Manager/Team
Description:	Software and hardware alternatives are reviewed during the overall project Initiation and Planning Stage and are used to formulate preliminary platform options. Use the project objectives, scope, and high-level requirements as the basis for conducting research and investigating resources.
Sample Questions:	The following is a list of sample questions that can be used to help investigate software and hardware alternatives and provide recommendations that fit within the budget and goals for the project. • Is the alternative feasible within time and resource 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. ◦ 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 investigating alternatives and will assist in completing the Enterprise Architecture (EA) Solution Assessment document. Once the EA Core Team develops their own instruction set for the Solution Document, these sections will be removed from the SEM. C.1.1 Investigate Software Alternatives C.1.2 Investigate Hardware Alternatives C.1.3 Formulate Platform Options
Touch Points:	The following touch points are involved in investigating alternatives: • Enterprise Architecture (EA) - Solution Assessment document • Contracts and Procurement – procurement related documents, including DIT-0153 Bid Information Sheet, DIT-0015b, Statement of Work • Office of Enterprise Security (OES) – Security liaison sign-off on Security Plan initiation

Task: C.1.1 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 State of Michigan and the commercial marketplace should be investigated. In the Initiation and 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 Department-wide 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 from the MDIT Bureau of Strategic Policy, 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, such as local units of government that use and interface with SOM systems.
- Adapt mainframe or minicomputer source code obtained from existing State of Michigan 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: C.1.2 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 State of Michigan and through the commercial marketplace. In the Initiation and 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
 - Underutilized hardware
 - New procurement
- Architecture compliance
- Current and future communications needs
- Computer security requirements of the system
- Volume of data
- Importance of data to the MDIT-Department-Agency missions
- 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 SOM 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: C.1.3 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 State of Michigan 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 Initiation and Planning Stage is revisited in the Functional Design Stage. At this time a final recommendation is developed by the project team, including the appropriate Infrastructure Services staff, and presented to the system owner. The system owner is responsible for making the final platform decision.

Work Product: Develop an EA Solutions Assessment for use in the Infrastructure Services Request process and the Hosting Solution Document. Place a copy of these documents in the Project File.

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

Appendix D – List of Acronyms

CASE	Computer-Aided Software Engineering
CMMI	Capability Maturity Model Integrated
COTS	Customized Off-The-Shelf
EA	Enterprise Architecture
FCA	Functional Configuration Audit
IT	Information Technology
LAN	Local area network
MDIT	Michigan Department of Information Technology
OES	Office of Enterprise Security
PCA	Physical Configuration Audit
PMM	Project Management Methodology
SCM	Software Configuration Management
SEI	Software Engineering Institute (at Carnegie-Mellon)
SEM	Systems Engineering Methodology
SOM	State of Michigan
SUITE	State Unified Information Technology Environment
SWT	Structured Walkthrough

Appendix E – Roles and Responsibilities

The following list of roles and responsibilities should be considered as a standard baseline for all projects. Your project may include more roles. The same person may fulfill more than one role and there may be multiple individual's serving in a role on a given project.

Business Sponsor – The individual from the client Agency with decision-making authority over a project or group of projects.

Business Subject Matter Expert (SME) – The individual who is the authority or expert on the business need being addressed by the Project.

Configuration Manager – The individual responsible for identifying and defining Configuration Items (CI's) in a system, recording and reporting the status of CI's and Requests for Change (RFC), and verifying the completeness and correctness of CI's. This role is also responsible for the build and deployment of CI's and configuration verification and auditing.

Database Administrator (DBA) – The individual responsible for the security and information classification of the shared data stored on a database system. The responsibility includes the design, definition and maintenance of the database.

Information Technology (IT) Analyst – The individual skilled in the development and definition of the techniques and technologies being used for the project.

Information Technology (IT) Developer – The individual whose responsibility is the development of the software for the project.

Information Technology (IT) Lead – The individual responsible for all aspects of the system design and construction. The IT Lead directs the development team.

Information Technology (IT) Subject Matter Expert (SME) – The individual who is the authority or expert on the technology or technologies being implemented for the project.

Lead Business Analyst – The individual responsible for analyzing the business needs of their Agency to help identify business problems and propose solutions.

MDIT Sponsor – The individual from MDIT with decision-making authority over a project or group of projects.

Project Manager – The individual responsible for all activities of a project. The Project Manager plans, directs, controls, administers, and regulates a project.

Quality Assurance Analyst – The individual who audits and approves project deliverables from a QA perspective. Reviews plans and deliverables for compliance with applicable standards. Provides guidance and assistance on process matters and defining standards. Primary focus is on defect prevention.

Quality Control Analyst – The individual responsible for checking the product or service after it has been developed. Primary focus is to find defects.

Security Analyst – The individual skilled in the development and definition of the security controls of the project environment.

Training Coordinator – The individual who is the key person and point of contact for all training required for the project.

Ultimately the Project Manager is responsible for assuring that all tasks, activities, and work products are complete and accurate. However, other roles will be responsible for completing some or most of the various SEM forms. Below is a list of the SEM work products and what roles, as defined above, are responsible for completing the SEM forms. ***Your project may include more roles and approvers.*** The Project Manager is responsible for the completion of all checklists.

Functional Role	Deliverable												
	Configuration Management Plan	Conversion Plan	Functional Design	Installation Plan	Maintenance Plan	Project Metrics Collection	Requirements Specification	Security Plan	System Design	Test Plan	Test Reports	Training Plan	Transition Plan
Project Manager	A	A	A	A	A	E, A	A	A	A	A	A	A	A
Business / Client Sponsor	A		I		A	I	A	A				A	
MDIT Sponsor	A				A	I		A					
Lead Business Analyst			A/E	A			A/E	A		A/E	E	A/E	C
Business SME			C	C			C	C	C	C	C	C	
IT Lead	C	A/E	A/E	A/E	E	C, I	I	A/E	A/E	A/E	A/E		A/E
IT SME		C	C	C			I	C	C				C
IT Analyst	I	C	I	C	I	C	C	C	E	C	C		C
IT Developer	I	I	I	I	I	C	I	I	C				I
Database Administrator	C	E	C	E	C		I	C	E	C			C
Configuration Manager	A/E				C								
Security Analyst	I	I	I	I	I		C	A/E	C	I		C	I
QA Analyst		C	I	C			I	I	C	E	E		C
QC Analyst		C	I	C			I	I	C	E	E		C
Training Coordinator							I					A/E	

Legend: E = responsible for execution (may be shared) C = must be consulted A = responsible for final approval I = must be informed