



The CTA System Engineering Methodology (SEM): Information Technology (IT) Systems

IT Systems Engineering is an interdisciplinary, comprehensive approach to solving complex IT system problems and satisfying acquirer/user requirements.

In this definition, “*interdisciplinary*” indicates that a variety of engineering and non-engineering skills are required to be brought to bear concurrently to assure the system solution incorporates all required attributes efficiently and effectively. The multi-disciplinary team concept, sometimes called the Integrated Product Development Team (IPT) is an integral part of the methodology described in this document. The specific makeup of an IPT is project dependent, but will typically require system architects, subject matter experts, network engineers, software engineers, test engineers, security specialists, simulation/modeling experts, human factors analysts, system administrators, etc. An approach is “*comprehensive*” if it is systematic and complete in defining the process steps and management actions necessary to take a system from conceptual development to successful test and deployment. “*Complex IT system problems*” include new systems or system enhancements which involve complexities associated with scale, performance and/or functional attributes not readily available “off-the-shelf”. “*Requirements*” reflect needs, wants, expectations of buyers and users: what a system is to accomplish (functionality), performance features (scalability, response times, reliability, security, etc.), price, life-cycle cost, schedule, and environmental constraints (reuse of legacy components, legal, budgetary, political, etc.).

The CTA SEM represents over 20 years of successful experience in providing IT support services for government agencies and reflects the results of continual refinements and enhancements as improved systems engineering methods, tools, and processes have been proven and as they have become available. We have captured the “lessons learned” from completion of hundreds of assignments involving complex systems and have incorporated them into the SEM. This “white paper” and its accompanying graphic provide an introduction and overview of the CTA SEM. (The graphic may be printed for easier viewing. Return to the Systems Engineering page and select SEM Graphic.)

Our experience spans all activities typically included in the full system engineering lifecycle from early concept development through system O&M. From this extensive contract experience base, the CTA SEM has evolved to provide a comprehensive framework to successfully guide the management of large-scale IT project developments/enhancements and provide engineering support services in all key areas of the lifecycle. These key areas include: the development of complete, consistent, testable system requirements (understanding the problem before designing the solution) which will meet the “real” operational requirements expected by users, followed by the development of an affordable, highly maintainable, testable solution architecture which will also meet exacting performance requirements such as real-time operations and high levels of system security, scalability, and systems reliability and availability.

Although our SEM is based on a standard lifecycle model, it reflects the true iterative nature of system evolutions and provides for early QA testing and configuration management of the results of all phases using a comprehensive, integrated set of CTA-developed and industry-standard techniques and tools, procedures, and products applicable to each of these phases. The standard lifecycle phases can be tailored to such standards as MIL-STD-498, ISO 9000, Navy Lifecycle Management Review Handbook, Federal Information Processing Standards (FIPS) 38 and 64, and Military-Standard 2167A and the SEI Capability Maturity Models (CMM). As such, the SEM has been used successfully to augment existing client lifecycle processes and methodologies driven by specific client needs. For example the following models have been used successfully within the context of the SEM:

- 1) Waterfall Model: The traditional top-down development approach; an ordered set of phases that are performed in sequence, starting with a complete set of system requirements;
- 2) Evolutionary/Incremental Development Model: The execution of successive increments, delivering the “must-have” requirements at each stage, adjusting requirements so as to deliver prioritized

capabilities at each stage. (This approach is called Dynamic Systems Development Method (DSDM) in Europe.) Approach uses structured rapid prototyping techniques and requires a high degree of user participation during development;

- 3) Reusable Components Model: This model begins with an assessment of reusable components available, and augments the system with new functionality, adjusting requirements whenever possible to capitalize on the reuse of existing, reliable components;
- 4) Technology Application Model: This model begins with an assessment of existing or emerging technology and involves adjusting requirements whenever possible to capitalize on the technology. The drive to maximize the use "COTS" is a good example;

As the information technology evolution driven by high performance embedded electronics, wireless technology and the Internet has progressed at an ever increasing pace, CTA has made a commitment to not only stay abreast of these changes but also lead the changes in some key technology areas. We are in the forefront of technologies enabling high performance real-time embedded applications, high levels of network and information systems security (INFOSEC), object-oriented software engineering, computer aided software engineering (CASE) and I (integrated)-CASE tools, rapid application development (RAD) methodologies, distributed system/database architectures, wireless communications, and the practical use artificial intelligence/knowledge-based systems.

The attached Exhibit portrays the entire SEM life cycle from concept development through system deployment. The upper half of the exhibit describes the top-level process flow through each of the phases and the analytical steps required to achieve the products of each phase. The bottom half of the CTA SEM Exhibit indicates the *Methodologies, Engineering Products, and Engineering Processes* that are used to address the engineering requirements during any given lifecycle phase. It is important to note, however, that we do not propose to use each and every one of these methodologies, products and processes listed in performance of a specific engagement. Rather, the CTA SEM represents an initial identification of a comprehensive toolkit of resources that, upon receipt of a specific engineering task, can be used to assemble a specific set of tools appropriate to the assignment. An important product of the early stages of the SEM as applied to a specific program is the development of the Systems Engineering Management Plan (SEMP) that describes how the fully integrated engineering effort, including processes, methodologies, tools will be executed over the specific development lifecycle.

Requirements Engineering

It is well established that the requirements definition and analysis phase of the development lifecycle is the most crucial: major government IT systems have ended in disaster because of the lack of a well-defined and **controlled** set of baseline requirements. ("Scope creep" can be as damaging as ill-defined, vague, un-testable "requirements".) The SEM reflects this fact by addressing in meticulous detail **functional** requirements (the "what" of system requirements), **performance** requirements (the "how" a system is required to perform; e.g. how responsive, how secure, how reliable, etc.) and most importantly, **operational** requirements (how the user will use the system, what he will "see", how he will control, how his decision-making will be supported, etc.). The SEM also reflects **constraints** that must be met by a new system: technical, interface, political, procedural, etc.

Functional operators and technical users help define the current operating environment and are integral in developing assumptions based on known alternatives and constraints. Our methodology involves the functional and technical users from the beginning. This ensures that we determine the "real problem" to be solved: e.g. what will be automated and what will remain manual operations, and the environmental constraints that may play regardless of the size and/or scope of the problem area.

A key product of this early phase is the **Concept of Operations**, recording detailed operational performance, functional and user interface requirements. This step includes the development of systems scenarios and the analysis of functional requirements relative to the capabilities and deficiencies of existing systems. This analysis may be through composition graphs, functional flow block diagrams, data flow diagrams, or other similar tools. Test cases and success criteria are developed from these scenarios. We document the "as is" state of the system, which includes current hardware, software, communications, assumptions, and constraints that are inherent in the current environment, including "non-functional" requirements such as power, volume, weight, dimensions, etc. We provide services in performance and

capacity modeling, planning for long-range technology evolution, human factors engineering, design of interfaces with external systems, and both design and integration of the hardware, software, and network components comprising the client's systems solution.

The ultimate objective of the requirements engineering phase is the definition/derivation and validation of functional, performance and operational requirements that will be the basis for detailed functional analysis (decomposition) and allocation to design components of the solution architecture. An essential stage in this process is the allocation of performance requirements to system functions, i.e., for each derived functional requirement, determine how well that function must be performed. This will require the establishment of technical "budgets" for timing, system resources, reliability/availability and security which will be allocated to system components to assure overall system performance meets the system-level performance requirements.

Systems Architecture. During this stage another key product is the development of architectural alternatives that meet the top-level requirements. Such solution alternatives could result from a formal requirements decomposition/allocation process ("system synthesis"), but frequently evolves from models of successful prior implementations of similar systems. We apply the appropriate technologies and methodologies such as system level "modular cohesion and coupling metrics" to establish "optimal" (e.g., highly maintainable) architectural alternatives to be compared with sub-optimal (but perhaps lower cost or risk) architectures which utilize legacy system components. Given the trend from custom system components to the use of COTS (commercial-off-the-shelf) components, we frequently establish solution alternatives comprised of "best-of-breed" COTS hardware, software, and network components to satisfy a given client's infrastructure needs. System requirements are allocated to elements of the architecture, defining interfaces between system components, and the functional and performance requirements to be met by the component.

We also ensure the system **infrastructure** is engineered to fully support the **functional and performance** demands of the software applications in the context of the **Concept of Operations**. Our systems design and engineering methodology provides for a complete documentation trail from system architecture and design descriptions, requirements specifications, and enterprise model descriptions to the lower-level design descriptions and software specifications.

Cost/Risk Analysis. Cost and economic analysis is an integral part of any program decision. We life cycle cost and compare the alternatives to a common set of prioritized client objectives and constraints. Costing technologies (e.g. see for example cost estimating models at <http://dacs.dtic.mil/techs/baselines/appendixB.html>), risk methodologies (see for example Risk Models for this purpose at <http://www.decisioneering.com>) and activity-based or target product costing are used to ensure accurate cost data. All models are customized to specific client requirements. Requirements generally include cost, reliability, performance, compatibility/interoperability, commonality, most economic use, security considerations, compliance with standards, ease of use, time constraints, or any other criteria prescribed for a particular project. We define the resources and technologies that are required by each of the alternatives and rank order the alternatives based on cost/benefit analysis.

Modeling. To support the evaluation of alternatives we develop a functional process model that includes cost, process/work flows, and the "as is" process model brought forward from the Concept of Operations phase. A variety of tools and technologies are used to produce the functional process model. These models may include physical prototypes, mathematical formulations, simulation models, breadboards, executable code frames, etc.

Documentation. Each process activity of the Requirements Analysis and Concept Development Phase is documented and the appropriate reports, plans, and documents are produced. These reports are either oral or written and contain a presentation of the functional/performance requirements; descriptions of the alternative solutions for satisfying the requirements; a comparison of the costs/benefits/risks associated with each alternative; assumptions and constraints; estimates of the development times and manpower/resource requirements; and recommendations and issues.

Technology, Product and/or Product Area-Evaluation. CTA works with hundreds of vendors who provide a variety of products and services. We are frequently called upon to help develop leading edge

architectures based on the future direction of specific technologies. We maintain an objective, independent perspective on the product marketplace, provide laboratory-based assessments and benchmarks when needed and otherwise help our clients make decisions about the use of existing, new, “best-of-breed”, or emerging technologies. Government-owned, COTS, and proprietary applications are examined to determine their applicability to the requirement. Tools such as “middleware”, object-oriented tools, wireless PDA, EDI/XML, security products, etc., are objectively examined to see if they can be applied to enhance performance, lower risk or expedite the developmental process. The product evaluation cycle results in a recommendation based on the assumptions and constraints that have been identified.

Demonstration Plan. The first step of concept demonstration is the development of a plan that articulates the baseline that the product or technology should be evaluated against and identifies the acceptable performance criteria. Included in the plan is a description of the operating environment (hardware, software, and communications constraints), site preparation requirements, the performance plan, and the expected results. Following the demonstration, an analysis is performed and compared against the baseline performance as identified in the Demonstration Plan. The results are documented and recommendations made based on the constraints identified during the Requirements Analysis.

Capacity Management. Failure to consider the performance capability needed by the underlying hardware and operating system environment jeopardizes the application’s ability to work on a timely basis. If hardware and throughput requirements are overlooked during system development, the project’s cost and schedule can be adversely affected. Consequently, these issues must be identified as early as possible in the development process to ensure that the selected hardware (including networks) environment will provide sufficient storage, response times, printing capacity and other performance requirements. Early identification is also required to ensure that adequate funds and time are set aside to allow for completion of the procurement cycle.

The process of identifying the capabilities of the target platform (defined herein as the necessary hardware, operating system, database management system, network operating software, graphical user interface, and other hardware and software elements needed for the application software and its related data to function as intended) requires coordination and monitoring with many other elements of the development process.

Estimation of capacity and performance factors must be based on the functional requirements document and the process modeling and data modeling design documents. The data used for capacity and performance estimating must be traceable to the design documents. The data model must be used to determine the quantity of data that must be stored and transmitted. The flows of data to be transmitted must be derived from the process model. Finally, the functional requirements document must be used to specify when the transactions will occur, who will make them, and where they will be made. Following implementation, the requirements documents are placed under configuration control and periodic test reports are generated to monitor capacity and performance factors.

Security and Survivability Engineering. The CTA Team approach to developing security and survivability plans is based on a planning methodology that emphasizes early assessment of the sensitivity and criticality of a system. This assessment includes identifying the system’s purpose and operating environment; the system’s criticality to mission performance; the sensitivity of the data being handled (privacy and classification protection requirements) by the system; and the management, operational, and technical controls to be applied. This ensures a level of security and redundancy effort commensurate with the importance of the system. The addition of adequate security and survivability capabilities to an operational system can be extremely difficult and expensive (if not impossible), demanding the early introduction of security and survivability engineering concepts in the development lifecycle.

High-level Systems Design

The Requirements Analysis and Specification process generates the Functional Requirements Definition (the Functional Description Document, in CTA SEM terms). This provides the system summary; the detailed characteristics of proposed/required system performance, functionalities, input/output, data, and

contingencies for failures; the design configuration, the operation, testing, development hardware, networking/communication and software environments; the security aspects; the cost factors; the software system lifecycle development plan, the QA plan, and the CM plan. The High-level Design process, using the results of the Analysis and Specification process, and design strategies such as OOD and complexity metric minimization, produces the Process Design, the Architectural Design, the Interface Design, the System/Subsystem Specification, and the Database Logical Design.

Our approach establishes the appropriate operational environment including the host platform, networking/communication, external interfaces, development, test, and operating environment. Processes are defined and system/software architectures are developed. Requirements are allocated to subsystems and programs and a system/subsystem design document is generated. High quality support is ensured by our focus on software QA and process improvement throughout the development lifecycle.

Database Design and Engineering methodologies used by CTA are tailored to meet specific client needs. Selection of the methodology depends on activity, system characteristics, architecture, and the client legacy migration strategy.

Detailed Design

We develop a Detailed Design Specification (DDS) for each element of the high-level architecture design, to include (but not limited to) all inputs/outputs, algorithms, performance requirements and data flows. Detailed design expands the system/subsystem design (data, functional, state, object orientation). We use the same representations of control, data, and structure used in functional requirements definition, system/subsystem specification, and database design. Detailed design complements and refines the higher-level design and produces a design from which coding specifications are developed. Detailed Action Diagrams (DAD) and/or Program Design Language (PDL) are used to convey this detailed design.

A Detailed Design Specification also provides details of input processing, checks performed on incoming data, handling of exceptions, required validity checks, data storage locations, and required processing. It focuses on readability, consistency, and clarity. CTA uses appropriate CASE tools (such as Oracle Designer, Forte, ADW, IEF) to develop DADs or PDL. This adds rigor and precision since the tools can identify and eliminate ambiguities and inconsistencies. If a code generator tool is used, we typically use the DADs as the tools can generate code directly from these detailed diagrams.

Where appropriate, we use Objected Oriented Design (OOD). Elements are represented by object diagrams and package/task specifications. A package is a collection of entities (variables, subprograms, type declarations, and exceptions) that can be used by other packages. The specification describes external appearance of the package. This includes types, exceptions, variables, names, parameters, and results. CTA uses applicable tools (such as PowerBuilder, Key/ADW, and IEF) to define package specifications, impose precision, and facilitate use of code generators in the coding phase.

Design “Walk-throughs”. Designs “walk-throughs” are performed to assess and test the validity and completeness of the proposed design. A walk-through is a peer review of a product and has been found to be an effective approach to detect design flaws early. It may take the form of a code inspection by other programmers, a technical review by team members, or a more formal group review. Walk-throughs verify and validate all system development lifecycle deliverables. Our system development lifecycle review methodology provides numerous checklists that form the basis for interim and final product reviews and are further tailored by team members to address unique concerns specific to task requirements.

CTA follows standard review formats to ensure the presented approach/design is reviewed and validated for consistency, completeness, traceability, feasibility, testability, and correctness and to identify potential problems early in the development process. Corrections at this level have less impact than when discovered later. Design walk-through is a standard CTA software development technique and is integral to the SW QA process.

Systems Development

System Development constitutes the planning and execution of the definition, design, implementation, integration, analyses, and control tasks, actions, and activities required to evolve the system from Client needs to system product and process solutions. Development applies to new developments, product improvements, and modifications, as well as any assessments needed to determine a preferred course of action for material solutions to identified needs, deficiencies, or problem reports. Our process is capable of beginning a full development or receiving a partial development and implementing the detailed design and conducting the testing necessary to achieve and demonstrate full functionality and required performance of the total system as well as subsystems and individual components. Since many developments are performed in the context of improving an existing system while maintaining current operations, independent development environments are frequently used, logically identical to the operational infrastructure to assure successful transition to the operational environment.

The system/subsystem specifications define the top-level design by identifying software components for the application; logically grouping the functional requirements; and establishing sub-system interfaces, inputs/outputs, data retention requirements, performance requirements, and operating environment. We use N² Analysis Charts, performance modeling, operations concept analysis, requirements traceability matrices, structure charts, and object-oriented diagrams to fully document the modification or new development.

Configuration management, requirements baseline controls and developer communications are key issues to be addressed during this phase.

System Integration, Verification and Validation

System Integration, Verification and Validation (SIV&V) integrates the components of the architecture at each level, **verifies** that the requirements for those components are met by testing against the specified technical requirements for that component, and **validates** that the requirements are complete and consistent with the higher level requirements of the system. Validation should occur as early as possible in the development lifecycle. Typically verification focuses on identifying and removing defects from the developed product, while validation focuses on demonstrating to the user that the product meets the system requirements.

This activity requires the following stages:

- 1) Defining the specific methods that will verify and validate the functional, performance and operational requirements, where each method identifies the requirement to be verified, the test procedure to be followed, and success criteria. We use the Requirements Correlation Matrix (RCM) to document and track a requirements verification technique for acceptance test, demonstration, analysis, and inspection for each requirement. We use the detailed design specifications as the basis for all test design activities. We follow distinct but complementary paths to design test data. Analysis of design specifications reveals test data values and sequencing and control information required to develop associated test procedures. CTA evolves the RCM in parallel with program design refinements. Early agreement on acceptance criteria for each requirement ensures testable requirement specifications and feasible test plans. The RCM maps detailed requirements to specific test cases and software components. If requirements change, the RCM identifies the test procedures and data affected. A report is generated to indicate the change to the test team. The test team modifies the test procedures accordingly.
- 2) Defining and establishing the test environment within which the SIV&V activity will be performed. This includes the test facility, simulation systems, test data sources, and special drivers or tools needed to drive the system under realistic operational conditions.
- 3) Obtaining customer approval of the SIV&V program prior to the start of test execution. (In fixed-price and time/cost constrained situations, getting approval of an acceptance test plan early in the lifecycle is mandatory.)

System Deployment. Deployment activities consist of installing and testing the system in the final operational environment, performing all needed data loading/conversion, conducting training in the operation of the system, conducting and documenting performance tests, making necessary software/hardware/configuration/ procedures modifications required as a result of these tests. We use automated test tools to help support the various test milestones during testing and to support ongoing testing activities associated with system maintenance.

Post Deployment Evaluations. Post Deployment activities consist of monitoring and analyzing existing operations, performance, and capacity to determine if the system requirements can be met by optimally tuning the system or by procuring additional hardware and software. CTA performs network system tuning by analyzing network performance and system operation data for protocol errors, performance bottlenecks, and network utilization. We provide on-going training to users on the new releases of software and hardware and any changes in operational procedures. We also provide both on-call and/or on-site system maintenance support as specified by the client.

Configuration Management Configuration management (CM) is the set of technical and administrative direction and surveillance actions taken to identify and document the functional and physical characteristics of a configuration item (identification), to control changes to a configuration item and its characteristics (control), to record and report change processing and implementation status (status accounting), and to verify that the configuration of hardware and software conforms to the current approved documentation (audit). CM transcends the development of administrative procedures and accountability schemes (although these are a vital part of the process). Viewed from a concurrent engineering perspective, CM includes those systems engineering analyses required to ensure that a system is properly managed and maintained throughout its lifecycle. This includes ensuring the requirements for CM are factored into the early stages of system development, that CM activities have been considered for the entire family of components and functions that comprise the system, and that the system is designed and modified in a balanced fashion (one that is consistent with cost, schedule, and performance characteristics and constraints). CM is integrally connected with systems engineering and must be considered throughout the lifecycle.