Glossary

Acceptance: takeholder function for agreeing that the designed system, as tested or otherwise evaluated by the stakeholders, is acceptable.

Acceptance Plan: how the qualification data will be used to determine that the real system is acceptable to the stakeholders.

Allocated Architecture: complete description of the system design, including the functional architecture allocated to the physical architecture; derived input/output; technology, system-wide, trade-off, and qualification requirements for each component; an interface architecture that has been integrated as one of the components; and complete documentation of the design and major design decisions.

Apportionment: requirements flowdown approach that spreads a system-level requirement among the system's components of the system, maintaining the same units.

Attainable: solutions exist within performance, cost, and schedule constraints.

Behavior Model: defines the control, activation, and termination of system functions that are needed to meet the performance requirements of the system.

Bipartite Graph (Digraph): graph (digraph) whose set of nodes can be partitioned into two sets A and *B* such that no edge connects a node in A to another node in A and, similarly, no edge connects a node in *B* to another node in *B*.

Black Box Testing: outputs are determined correct or incorrect based upon inputs; inner workings of the module are ignored. Both positive and negative testing have to be employed.

The Engineering Design of Systems: Models and Methods, Second Edition. By Dennis M. Buede

Cartesian product of two sets, *A* x *B*: set of all possible ordered pairs of those two sets.

Centralized Architecture: architecture with a central location for the execution of the transformation and control functions of the system.

Client–Server Architecture: architecture that distinguishes between client processes (requestors) and server processes (task completors).

Comparable: pertaining to requirements, the relative necessity of the requirements is included.

Complete: pertaining to requirements: (a) everything the system is required to do throughout the system's life cycle is included; (b) responses to all possible (realizable) inputs throughout the system's life cycle are defined; (c) the document is defined clearly and self-contained; and (d) there are no "to be defined" (TBD) or "to be reviewed" (TBR) statements. Completeness is a desired property but cannot be proven at the time of requirements development, or perhaps ever.

Component: subset of the physical realization (and the physical architecture) of the system to which a subset of the system's functions have been (will be) allocated. A component could be the integration of hardware and software, a specific piece of hardware, a specific segment of the system's software, a group of people, facilities, or a combination of all of these.

Conceptual Validity: correspondence between the stakeholders' needs and the operational concept.

Concise: pertaining to requirements, no unnecessary information is included in the requirement.

Configuration Items: lowest level components in the physical architecture.

Consistent: pertaining to requirements (a) internal — no two subsets of requirements conflict and (b) external — no subset of requirements conflicts with external documents from which the requirements are traced.

Context of a System: set of entities that can impact the system but cannot be impacted by the system.

Correct: pertaining to requirements, what the system is in fact required to do.

Cost Requirement: requirement addressing the payment of money during the appropriate life-cycle phase for the system in question to be useful.

Data Model: defines the relationships among the inputs and outputs of a system.

Deadlock: undesired state of the system in which activity ceases and throughput is nonexistent. Deadlock can occur for two reasons: contention over resources and waiting for a communication.

Decentralized Architecture: architecture with multiple, specific locations at which the same or similar transformational or control functions are performed.

Decision: irrevocable allocation of resources to affect some chosen change or the continuance of the status quo.

Definitive Model: addresses the question of how an entity should be defined.

Descriptive Model: attempts to predict answers to questions for which the truth may or may not be obtained in the future.

Design: preliminary activity that has the purpose of satisfying the needs of the stakeholders. Design begins in the mind of the lead engineer but has to be transformed into models employing visual formats in a highly skilled manner for success to be achieved.

Design Independent: pertaining the requirements, each requirement does not specify a particular solution or a portion of a particular solution.

Design Validity: congruence between the Originating Requirements Document (ORD) and the derived requirements.

Directed Graph or Digraph: pair of sets, $V(G)$ and $E(G)$. $V(G)$ $\{n_1, n_2, \ldots, n_N\}$ is the set of vertices or nodes. $V(G)$ is a finite, non empty set. $E(G) = e_{ij}$ is a subset of $V \times V$ or *ordered* pairs of nodes. e_{ij} is said to be from n_i to n_j. $E(G)$ may be empty.

Distributed Architecture: architecture in which there are two or more autonomous processors connected by a communications interface and running a distributed operating system.

Early Validation: determination that the right problem is being defined at the current level of abstraction given the validity of the problem definition at a higher level of abstraction.

Engineering: discipline for transforming scientific concepts into cost-effective products through the use of analysis and judgment.

Engineering of a System: engineering discipline that develops, matches, and trades off requirements, functions, and alternate system resources to achieve a cost-effective, life-cycle balanced product based upon the needs of the stakeholders.

Entity–Relationship Diagrams: model of the data structure or relationships between data entities.

Equivalence: simple requirements flowdown approach that causes the component requirement to be the same as the system requirement.

Error: subset of the system state that may lead to a failure. The system can monitor its own state, so errors are observable in principle.

External Interface Requirements: limitations placed upon the receipt of inputs and transmission of outputs by the interfaces of the external systems.

External Systems Diagram: model of the interaction of the system with other (external) systems in the relevant contexts, thus providing a definition of the system's boundary in terms of the system's inputs and outputs.

Failure: deviation in behavior between the system and its requirements. Since the system does not maintain a copy of its requirements, a failure is not observable by the system.

Fault: defect in the system that can cause an error. Faults can be permanent (e.g., a failure of system component that requires replacement) or temporary due to either an internal malfunction or external transient.

Feedback and Control: comparison of the actual characteristics of an output with desired characteristics of that output for the purpose of adjusting the process of transforming inputs into that output.

Figure of Merit (FOM): describes a specific system property or attribute for a given environment and context; an FOM is measured within the system. Also called a measure of performance (MOP).

Function (Mathematical): binary relation from A to B such that every element of A is mapped one and only one element of B.

Function (Engineering): process that transforms inputs into outputs.

Functional Architecture: (a) logical architecture that defines what the system must do, a decomposition of the system's top-level function. This very limited definition of the functional architecture is the most common and is represented as a directed tree. (b) Logical model that captures the transformation of inputs into outputs using control information. This definition adds the flow of inputs and outputs throughout the functional decomposition. (c) Logical model of a functional decomposition plus the flow of inputs and outputs, to which input/output requirements have been traced to specific functions and items (inputs, outputs, and controls).

Functional Requirements: the two to seven functions that are the first-level decomposition of the system's function.

Fundamental Objective: aggregation of the essential set of objectives that summarizes the current decision context and is yet relevant to the evaluation of the options under consideration.

Functionality: set of functions required to produce a particular output. *Simple functionality* is an *ordered* sequence of functional processes that operates on a single input to produce a specific output. Note there may be many inputs required to produce the output in question, but this simple functionality is only related to one of the inputs. *Complete functionality* is a complete set of coordinated processes that operate on all of the necessary inputs for producing a specific output.

Fundamental Objectives Hierarchy: subdivision of the fundamental objective into value objectives that more meaningfully define the fundamental objective, thereby forming a value structure.

Graph, G: a pair of sets, $V(G)$ and $E(G)$. $V(G) = \{n_1, n_2, \ldots, n_N\}$ is the set of vertices or nodes. $E(G) = \{e_{ij}\} \subseteq V(G) \times V(G)$ is a relation that defines the set of edges that are unordered, not necessarily distinct pairs of nodes. $V(G)$

is a finite, nonempty set. $E(G)$ may be empty and is a subset of the Cartesian product of $V(G)$ with itself.

Hardware Redundancy: use of extra hardware to enable the detection of errors as well as to provide additional operational hardware components after errors have occurred. *Passive hardware redundancy* masks or hides the occurrence of errors rather than detecting them; recovery is achieved by having extra hardware available when needed. *Active hardware redundancy* attempts to detect errors, confine damage, recover from the errors, and isolate and report the fault.

ICOMs: the inputs, controls, outputs, and mechanisms of a function in IDEF0.

IDEF0: IDEF acronym comes from the U.S. Air Force's Integrated Computer-Aided Manufacturing (ICAM) program that began in the 1970s. IDEF is a complex acronym that stands for ICAM Definition. The number, 0, is appended because this modeling technique was the first of many techniques developed as part of this program. More recently, the U.S. Department of Commerce [the National Institute of Standards and Technology (NIST)] has issued Federal Information Processing Standard (FIPS) publication 183 that defines the IDEF0 language and renames the acronym, *I*ntegrated *De*finition for *F*unction modeling.

Information Redundancy: addition of extra bits of information to enable error detections using special codes.

Input/Output Requirements: requirements about sets of acceptable inputs and outputs, trajectories of inputs to and outputs from the system, interface constraints imposed by the external systems, and eligibility functions that match system inputs with system outputs for the life-cycle phase of interest. This category is partitioned into four subsets: (a) inputs, (b) outputs, (c) external interface constraints, and (d) functional requirements.

Input/Output Trace: a time line associated with each major actor (our system and other systems) in the scenario. The systems involved are listed across the top of the diagram with the time lines running vertically down the page under each of the systems. Time moves from top to bottom in an input/output trace; the system of concern is highlighted with a bold label and heavier line. Interactions involving the movement of data, horizontal arcs from the originating system to the receiving system designate energy or matter among systems. A label is shown just above each arc to describe the data or item being conveyed. Double-headed arcs are permissible to represent dialog in a compact manner. Having two or more arcs in quick succession is also common to illustrate that the same item is being transmitted from one system to multiple systems or multiple systems are potentially transmitting the same item to one system.

Input Requirements: inputs the system must receive and any performance or constraint aspects of each.

Integration: process of assembling the system from its components, which must be assembled from their configuration items (CIs).

Interface: connection for hooking to another system (an external interface) or for hooking one system component to another (an internal interface). The interface of a system contains both a logical element and a physical element (or link) that are responsible for carrying items (electromechanical energy or information) from one component or system to another.

Items: inputs that are received by the system, the outputs that are sent by the system to other systems, and the inputs that are generated internally to the system and sent to other parts of the system to assist in the transformation process for which the system is responsible.

Life Cycle: begins with the gleam in the eyes of the users or stakeholders, is followed by the definition of the stakeholders' needs by the systems engineers, includes developmental design and integration, goes through production and operational use, usually involves refinement, and finishes with the retirement and disposal of the system.

Livelock: undesired state of the system in which resources are being routed in cycles (oscillating) while waiting for the proper allocation of resources to enable the completion of necessary activities; unfortunately the proper allocation of resources is never achieved and the system cycles continuously, never reaching the desired outputs.

Manufacturing: using resources to perform operations on materials to produce products.

Measure of Effectiveness (MOE): variable that describes how well a system carries out a task or set of tasks within a specific context; an MOE is measured outside the system for a defined environment and state of the context variables.

Measure of Performance (MOP): variable that describes a specific system property or attribute for a given environment and context; a MOP is measured within the system.

Mental Model: abstraction of thought.

Mission Requirements: requirements that relate to objectives of the stakeholders that are defined in the context of the supersystem, not the system itself.

Mode of a System: distinct operating capability of the system during which some or all of the system's functions may be performed to a full or limited degree.

Model: any incomplete representation of reality, an abstraction. The *essence* of a *model* is the question or set of questions that the model can reliably answer for us.

Modifiable: pertaining to requirements, changes that can be made easily, consistently (free of redundancy), and completely.

Morphological Box: matrix in which the columns (or rows) represent the components in the generic physical architecture. The boxes in a given column (or row) then represent alternate choices for fulfilling that generic component.

Multiattribute Value Analysis: quantitative method for aggregating a stakeholder's preferences over conflicting objectives to find the alternative with the highest value when all objectives are considered.

Normative Model: model that addresses how individuals or organizational entities ought to think about a problem and guide decision making.

Objectives Hierarchy: hierarchy of objectives that are important to the system's stakeholders in a value sense; that is, the stakeholders would (should) be willing to pay to obtain increased performance (or decreased cost) in any one of these objectives. It is also the definition of the natural subsets of the fundamental objective into a collection of performance requirements.

Observance Requirement: requirement stating how the estimates (qualification data) for each input/output and system-wide requirement will be obtained. Typically one of the four major qualification methods (test, analysis and simulation, inspection, or demonstration) is assigned to each input/output and system-wide requirement.

Open Architecture: architecture in which the hardware and software interfaces are sufficiently well defined that additional resources can be added to the system with little or no adjustment.

Operational Concept: vision for what the system is (in general terms), a statement of mission requirements, and a description of how the system will be used. The shared vision is based on the perspective of the system's stakeholders of how the system will be developed, produced, deployed, trained, operated and maintained, refined, and retired to overcome some operational problem and achieve the stakeholders' operational needs and objectives. The mission requirements are stated in terms of measures of effectiveness. The operational concept includes a collection of scenarios (one or more for each group of stakeholders in each relevant phase of the system's life cycle).

Operational Validity: matching of the capabilities of the designed system to the operational concept; this naturally occurs late in the integration phase after the designed system has been verified.

Output Requirements: requirements that state what outputs the system must produce and any performance aspects.

Overlap in the Functional Architecture: redundancy in functionality that is not needed to achieve additional performance.

Partition on a Set A: collection P of disjoint subsets of A whose union is A.

Performance Analysis: analysis for the purpose of discovering the range of performance that can be expected from a specific design or a set of designs that are quite similar.

Performance Requirement: requirement defined on some index that establishes a range of acceptable performance from a minimum acceptable threshold to a design goal.

Physical Architecture: resources for every function identified in the functional architecture. The *generic physical architecture* is a description of the partitioned elements of the physical architecture without any specification of the performance characteristics of the physical resources that comprise each element (e.g., central processing unit). An *instantiated physical architecture* is a generic physical architecture to which complete definitions of the performance characteristics of the resources have been added.

Physical Model: representation of an entity in three-dimensional space. A physical model can be divided into full-scale mock-up, subscale mock-up, breadboard, and electronic mock-up.

Power Set of Set A: set of all sets that are subsets of A.

Process Model: model that defines the functional decomposition of the system function and the flow of inputs and outputs for those functions.

Prototype: physical model of the system that ignores certain aspects of the system, glosses over other aspects, and is fairly representative of a third segment of aspects of the system. The prototype can range from a subscale model of the system to a paper display (storyboard) of the user interface of the system.

Qualification: process of verifying and validating the system design and then obtaining the stakeholders' acceptance of the design.

Qualification Methods: inspection, analysis and simulation, instrumented test, and demonstration.

Qualification Requirements: requirements that address the needs to qualify the system as being designed right, the right system, and an acceptable system. There are four primary elements: (a) observance: to state which qualification data for each input/output and system-wide requirement will be obtained by (i) demonstration, (ii) analysis and simulation, (iii) inspection, or (iv) instrumented test; (b) verification plan: to state how the qualification data will be used to determine that the real system conforms to the design that was developed; (c) validation plan: to state how the qualification data will be used to determine that the real system complies with the originating performance, cost and trade-off requirements; and (d) acceptability: to state how the qualification data will be used to determine that the real system is acceptable to the stakeholders.

Qualitative Model: model that provides symbolic, textual, or graphic answers. Symbolic models are based on logic or set theory. Textual models are based

on verbal descriptions. Graphical models use either elements of mathematical graph theory or simply artistic graphics to represent a hierarchical structure, the flow of items or data through a system's functions, or the dynamic interaction of the system's components.

Quantitative Model: model that provides answers that are numerical; these models can be either analytic, simulation, or judgmental models.

Regression Testing: retesting a portion of the system after a change has been made to ensure that new problems were not introduced.

Relation (Binary): relation that relates elements of A to elements of B and is a subset, R, of $A \times B$.

Relation (Unary) on a Set A: relation that relates elements of A to itself and is a subset, R, of $A \times A$.

Requirements Flowdown: derivation of requirements from one level of the operational architecture for a lower level of the architecture. A requirements flowdown includes three approaches: apportionment, equivalence, and synthesis.

Requirements Statements: defines the needs and objectives of stakeholders.

Requirements Validity: correspondence between the operational concept and the originating requirements.

Risk: combination of the probability of an event occurring and the significance of the consequence of the event occurring.

Risk Analysis: analysis done early in the development process to examine the ability of the divergent concepts to perform up to the needed level of performance across a wide range of operational scenarios. At this time there remains substantial uncertainty about the stakeholders' needs, the state of technology under consideration, and the details of the operational architecture.

Risk Avoidance: selection of the low-risk alternative; unfortunately what seems to be low risk intuitively is high risk in some cases.

Risk Management: use of hedging strategies; a hedging strategy is the maintenance of fallback options in case a riskier option fails.

Risk Transference: transfer of risk to others, an example being the purchase of insurance.

Scenario: defines how the system will respond to inputs from other systems in order to produce a desired output. Included in each scenario are the relevant inputs to and outputs from the system and the other systems that are responsible for those inputs and outputs. The scenario should not describe how the system is processing inputs to produce outputs; rather, the scenario focuses on the exchange of inputs and outputs by the system with other systems.

Schedule Requirement: requirement addressing a timing issue for the relevant system for the phase of life cycle in question.

Semantics: study of relationships between signs and symbols and what they represent.

Set: a collection of well-defined objects called elements or members.

Shortfall in the Functional Architecture: absence of a functionality that is required to produce a desired output from one or more inputs.

Software Redundancy: use of multiple versions of the same software functionality to provide multiple operational software components in the event of a software failure.

Specification: collection of requirements that completely define the constraints and performance requirements for a specific physical entity that is part of the system.

Stakeholder: owner and/or bill payer, developer, producer or manufacturer, tester, deployer, trainer, operator, user, victim, maintainer, sustainer, product improver, and decommissioner. Each stakeholder has a significantly different perspective of the system and the system's requirements.

Stakeholders' Requirements: statements by the stakeholders about the system's capabilities that define the constraints and performance parameters within which the system is to be designed. These stakeholders' requirements focus on the boundary of the system in the context of these mission requirements, are written in the stakeholders' language, are produced in conjunction with the stakeholders of the system, and are based upon the operational needs of these stakeholders.

Stakeholders' Requirements Document (StkhldrsRD): document that contains the stakeholders' requirements. Sometimes called the Originating Requirements Document (ORD) or Operational Requirements Document.

Starvation: undesired state of the system that occurs when a function needs a particular resource for execution, but the resource is always allocated to other functions due to a poorly designed resource assignment algorithm.

State of the System: static snapshot of the set of metrics or variables needed to describe fully the system's capabilities to perform the system's functions.

Suitability Requirements: requirements that address quality concerns of a system and are system-wide in scope. Examples are availability and safety.

Surge or Race: undesired state of the system that occurs in relatively uncontrolled systems when components are competing with each other to perform a task.

Syntax: way in which words are put together to form phrases and sentences.

Synthesis: requirements flowdown approach for those situations in which the system-level requirement is comprised of complex contributions from the components, causing the component requirements that are flowed down

from the system to be based upon some analytic model. The derived requirements for each component will have significantly different units than the system-level requirement had.

System: set of components (subsystems, segments) acting together to achieve a set of common objectives via the accomplishment of a set of tasks.

System Context: set of entities that can impact the system but cannot be impacted by the system.

System (Human-Designed):

- specially defined set of segments (hardware, software, physical entities, humans, facilities) acting as planned,
- via a set of interfaces, which are designed to connect the components,
- to achieve a common mission or fundamental objective (i.e., a set of specially defined objectives),
- subject to a set of constraints,
- through the accomplishment of a predetermined set of functions.

System Requirements: translation (or derivation) of the originating requirements into engineering terminology.

System Requirements Document (SysRD): document that contains the system requirements.

System Task or Function: set of functions that must be performed to achieve a specific objective.

Systems (External of a System): set of entities that interact with the system via the system's external interfaces.

Technology and System-wide Requirements: constraints and performance index thresholds that are placed upon the physical resources of the system. This category can be partitioned into four subsets: (a) technology, (b) suitability and quality issues, (c) cost for the relevant system (e.g., development cost, operational cost), and (d) schedule for the relevant life-cycle phase (e.g., development time period, operational life of the system).

Technology Requirement: constraints for the engineering creativity and should result from the other requirements if they are justifiable. These requirements are usually justified on the basis of interoperability or compatibility with an existing product line, which ultimately should be reflected in cost savings.

Time Redundancy: use of extra processing when time is available to perform the same computation multiple times with a single hardware and software combination and then compare the results.

Trade Study: analysis that focuses on finding ways to improve the system's performance on some highly important objective while maintaining the system's capability in other objectives.

Tree: graph, G, with no loops in which there is a unique, simple (no loops), nondirected path (or semipath in the case of a digraph) between each pair of nodes. A *rooted tree* is a tree in which there is a designated "root" node. In a graph, the root node must have a degree of 1. In a directed tree, the root node must have no parents, or an in degree of 0. A *directed tree* is a rooted tree in which there is a (directed) path from the root to every other node.

Traceable: pertaining to requirements, each derived requirement must be traceable to an originating requirement via some unique name or number.

Traced: pertaining to requirements, each requirement is traced to some document or statement of the stakeholders.

Trade-off Requirements: algorithms for comparing any two alternate designs on the aggregation of cost and performance objectives. These algorithms can be divided into (a) performance trade offs, (b) cost trade offs, and (c) cost–performance trade offs.

Unambiguous: pertaining to requirements, every requirement has only one interpretation.

Understandable: pertaining to requirements, interpretation of each requirement is clear.

Unique: pertaining to requirements, those that are not overlapping or redundant with other requirements.

Usability: includes ease of learning (learnability), ease of use (efficiency), ease of remembering (memorability), error rate, and subjectively pleasing (satisfaction).

Usability Testing: obtaining samples of users and eliciting the reactions of these users about their needs and desires as they interact with prototypes.

Validation: process of determining that the systems engineering process has produced the *right system*, based upon the needs expressed by the stakeholder.

Validation Plan: how the qualification data will be used to determine that the real system complies with the originating requirements.

Verifiable: finite, cost-effective process has been defined to check that the requirement has been attained.

Verification: matching of *Configuration Items* (CIs), components, subsystems, and the system to their corresponding requirements to ensure that each has been *built right*.

Verification Plan: how the qualification data will be used to determine that the real system conforms to the design that was developed.

White Box Testing: inner workings of the module are examined as part of the testing to ensure proper functioning. Usually used at the CI level of testing; this method becomes impractical at the system level.