System Requirements

From SEBoK
System Requirements

Lead Authors: *Alan Faisandier, Garry Roedler,* ***Contributing Author:*** *Richard Turner, Rick Adcock, Ariela Sofer*

System requirements are all of the requirements at the *system level* that describe the functions which the system as a whole should fulfill to satisfy the stakeholder needs and requirements, and are expressed in an appropriate combination of textual statements, views, and non-functional requirements; the latter expressing the levels of safety, security, reliability, etc., that will be necessary.

System requirements play major roles in systems engineering, as they:

- Form the basis of system architecture and design activities.
- Form the basis of system integration and verification activities.
- Act as reference for validation and stakeholder acceptance.
- Provide a means of communication between the various technical staff that interact throughout the project.

Elicitation of stakeholder requirements starts in Concept Definition and will be initially developed through interview and mission analysis. System requirements are considered in detail during System Definition. Neither can be considered complete until consistency between the two has been achieved, as demonstrated by traceability, for which a number of iterations may be needed.

Contents

Definition and Purpose of Requirements

A requirement is a statement that identifies a product or processes operational, functional, or design characteristic or constraint, which is unambiguous, testable, or measurable and necessary for product or process acceptability (ISO 2007).

To avoid confusion in the multitude of terms pertaining to *requirements*, consider the following classifications:

- **Process Role or State**: The role the requirement plays in the definition process; for instance, its position in the system block (e.g. translated, derived, satisfied) or its state of agreement (e.g. proposed, approved, cancelled).
- **Level of Abstraction**: The level within the definition process that the requirement stands; for instance, stakeholder requirement, system requirement, system element requirement.
- **Type of Requirement**: The nature of the requirement itself; for instance, functional, performance, constraint, etc.

Any single requirement may simultaneously be in a particular state, at a particular level of abstraction, and of a particular type. For additional explanations about differences between the types of requirements, refer to (Martin 1997, Chapter 2).

Principles Governing System Requirements

Relationship to Stakeholder Requirements and Logical Architecture

A set of stakeholder requirements are clarified and translated from statements of need into *engineering-oriented* language in order to enable proper architecture definition, design, and verification activities that are needed as the basis for system requirements analysis.

The system requirements are based around identification and synthesis of the functions required of any solution system associated with performance and other quality measures and provide the basis for the assessment of candidate solutions and verification of the completed system. The system requirements are expressed in technical language that is useful for architecture and design: unambiguous, consistent, coherent, exhaustive, and verifiable. Of course, close coordination with the stakeholders is necessary to ensure the translation is accurate and traceability is maintained. This results in a set of system functions and requirements specifying measurable characteristics which can form the basis for system realization.

The logical architecture defines system boundary and functions, from which more detailed system requirements can be derived. The starting point for this process may be to identify functional requirements from the stakeholder requirements and to use this to start the architectural definition,

or to begin with a high-level functional architecture view and use this as the basis for structuring system requirements. The exact approach taken will often depend on whether the system is an evolution of an already understood product or service, or a new and unprecedented solution (see Synthesizing Possible Solutions). However, when the process is initiated it is important that the stakeholder requirements, system requirements, and logical architecture are all complete, consistent with each other, and assessed together at the appropriate points in the systems life cycle model.

Traceability and the Assignment of System Requirements during Architecture and Design

Requirements traceability provides the ability to track information from the origin of the stakeholder requirements, to the top level of requirements and other system definition elements at all levels of the system hierarchy (see Applying Life Cycle Processes). Traceability is also used to provide an understanding as to the extent of a change as an input when impact analyses are performed in cases of proposed engineering improvements or requests for change.

During architecture definition and design, the assignment of requirements from one level to lower levels in the system hierarchy can be accomplished using several methods, as appropriate - see Table 1.

Table 1. Assessment Types for a System Requirement. (SEBoK Original)

Assignment Type for a System Requirement	Description
Direct Assignment	The system requirement from the higher level is directly assigned to a system or a system element for a lower level (e.g. the color used to paint visible parts of the product).
Indirect Assignment (Simply Decomposed)	The system requirement is distributed across several systems or system elements and the sum of a more complex calculation for distribution is equal to the requirement of higher level (e.g. a mass requirement, power distribution, reliability allocation, etc.) with sufficient margin or tolerance. A documented and configuration-managed "assignment budget" for each assignment must be maintained.
Indirect Assignment (Modeled and Decomposed)	The system requirement is distributed to several systems or system elements using an analysis or mathematical modeling technique. The resulting design parameters are assigned to the appropriate systems or system elements (with appropriate margins). For example, in the case of a radar detection requirement that is being analyzed, these lower-level parameters for output power, beam size, frequencies, etc. will be assigned to the appropriate hardware and software elements. Again, the analysis (or model) must be documented and configuration-managed.
Derived Requirement (from Design)	Such system requirements are developed during the design activities as a result of the decision of the design team, not the stakeholder community. These requirements may include the use of commercial-off-the-shelf (COTS) items, existing systems or system elements in inventory, common components, and similar design decisions in order to produce a "best value" solution for the customer. As such, these derived requirements may not directly trace to a stakeholder requirement, but they do not conflict with a stakeholder requirement or a constraint.

Classification of System Requirements

Several classifications of system requirements are possible, depending on the requirements definition methods and/or the architecture and design methods being applied. (ISO 2011) provides a classification which is summarized in Table 2 (see references for additional classifications).

Table 2. Example of System Requirements Classification. (SEBoK Original)

Types of System Requirement	Description
Functional Requirements	Describe qualitatively the system functions or tasks to be performed in operation.
Performance Requirements	Define quantitatively the extent, or how well and under what conditions a function or task is to be performed (e.g. rates, velocities). These are quantitative requirements of system performance and are verifiable individually. Note that there may be more than one performance requirement associated with a single function, functional requirement, or task.
Usability Requirements	Define the quality of system use (e.g. measurable effectiveness, efficiency, and satisfaction criteria).
Interface Requirements	Define how the system is required to interact or to exchange material, energy, or information with external systems (external interface), or how system elements within the system, including human elements, interact with each other (internal interface). Interface requirements include physical connections (physical interfaces) with external systems or internal system elements supporting interactions or exchanges.
Operational Requirements	Define the operational conditions or properties that are required for the system to operate or exist. This type of requirement includes: human factors, ergonomics, availability, maintainability, reliability, and security.
Modes and/or States Requirements	Define the various operational modes of the system in use and events conducting to transitions of modes.
Adaptability Requirements	Define potential extension, growth, or scalability during the life of the system.
Physical Constraints	Define constraints on weight, volume, and dimension applicable to the system elements that compose the system.
Design Constraints	Define the limits on the options that are available to a designer of a solution by imposing immovable boundaries and limits (e.g., the system shall incorporate a legacy or provided system element, or certain data shall be maintained in an online repository).
Environmental Conditions	Define the environmental conditions to be encountered by the system in its different operational modes. This should address the natural environment (e.g. wind, rain, temperature, fauna, salt, dust, radiation, etc.), induced and/or self-induced environmental effects (e.g. motion, shock, noise, electromagnetism, thermal, etc.), and threats to societal environment (e.g. legal, political, economic, social, business, etc.).
Logistical Requirements	Define the logistical conditions needed by the continuous utilization of the system. These requirements include sustainment (provision of facilities, level support, support personnel, spare parts, training, technical documentation, etc.), packaging, handling, shipping, transportation.
Policies and Regulations	Define relevant and applicable organizational policies or regulatory requirements that could affect the operation or performance of the system (e.g. labor policies, reports to regulatory agency, health or safety criteria, etc.).
Cost and Schedule Constraints	Define, for example, the cost of a single exemplar of the system, the expected delivery date of the first exemplar, etc.

Requirements Management

Requirements management is performed to ensure alignment of the system and system element

requirements with other representations, analyses, and artifacts of the system. It includes providing an understanding of the requirements, obtaining commitment, managing changes, maintaining bi-directional traceability among the requirements and with the rest of the system definition, and alignment with project resources and schedule.

There are many tools available to provide a supporting infrastructure for requirements management; the best choice is the one that matches the processes of the project or enterprise. Requirements management is also closely tied to configuration management for baseline management and control. When the requirements have been defined, documented, and approved, they need to be put under baseline management and control. The baseline allows the project to analyze and understand the impact (technical, cost, and schedule) of ongoing proposed changes.

Process Approach

Purpose and Principle of the Approach

The purpose of the system requirements analysis process is to transform the stakeholder, user-oriented view of desired services and properties into a technical view of the product that meets the operational needs of the user. This process builds a representation of the system that will meet stakeholder requirements and that, as far as constraints permit, does not imply any specific implementation. It results in measurable system requirements that specify, from the supplier’s perspective, what performance and non-performance characteristics it must possess in order to satisfy stakeholders' requirements (ISO 2015).

Activities of the Process

Major activities and tasks during this process include:

1. Analyzing the stakeholder requirements to check completeness of expected services and operational scenarios, conditions, operational modes, and constraints.
2. Defining the system requirements and their rationale.
3. Classifying the system requirements using suggested classifications (see examples above).
4. Incorporating the derived requirements (coming from architecture and design) into the system requirements baseline.
5. Establishing the upward traceability with the stakeholder needs and requirements.
6. Establishing bi-directional traceability between requirements at adjacent levels of the system hierarchy.
7. Verifying the quality and completeness of each system requirement and the consistency of the set of system requirements.
8. Validating the content and relevance of each system requirement against the set of stakeholder requirements.
9. Identifying potential risks (or threats and hazards) that could be generated by the system requirements.
10. Synthesizing, recording, and managing the system requirements and potential associated risks.
11. Upon approval of the requirements, establishing control baselines along with the other system definition elements in conjunction with established configuration management practices.

Checking Correctness of System Requirements

System requirements should be checked to gauge whether they are well expressed and appropriate. There are a number of characteristics that can be used to check system requirements, such as standard peer review techniques and comparison of each requirement against the set of requirements characteristics, which are listed in Table 2 and Table 3 of the "Presentation and Quality of Requirements" section (below). Requirements can be further validated using the

requirements elicitation and rationale capture described in the section "Methods and Modeling Techniques" (below).

Methods and Modeling Techniques

Requirements Elicitation and Prototyping

Requirements elicitation requires user involvement and can be effective in gaining stakeholder involvement and buy-in. Quality Function Deployment (QFD) and prototyping are two common techniques that can be applied and are defined in this section. In addition, interviews, focus groups, and Delphi techniques are often applied to elicit requirements.

QFD is a powerful technique to elicit requirements and compare design characteristics against user needs (Hauser and Clausing 1988). The inputs to the QFD application are user needs and operational concepts, so it is essential that the users participate. Users from across the life cycle should be included to ensure that all aspects of user needs are accounted for and prioritized.

Early prototyping can help the users and developers interactively identify functional and operational requirements as well as user interface constraints. This enables realistic user interaction, discovery, and feedback, as well as some sensitivity analysis. This improves the users' understanding of the requirements and increases the probability of satisfying their actual needs.

Capturing Requirements Rationale

One powerful and cost-effective technique to translate stakeholder requirements to system requirements is to capture the rationale for each requirement. Requirements rationale is merely a statement as to why the requirement exists, any assumptions made, the results of related design studies, or any other related supporting information. This supports further requirements analysis and decomposition. The rationale can be captured directly in a requirements database (Hull, Jackson, and Dick 2010).

Some of the benefits of this approach include:

- **Reducing the total number of requirements** - The process aids in identifying duplicates. Reducing requirements count will reduce project cost and risk.
- **Early exposure of bad assumptions**
- **Removes design implementation** - Many poorly written stakeholder requirements are design requirements in disguise, in that the customer is intentionally or unintentionally specifying a candidate implementation.
- **Improves communication with the stakeholder community** - By capturing the requirements rationale for all stakeholder requirements, the line of communication between the users and the designers is greatly improved. (Adapted from Chapter 8 of (Hooks and Farry 2000)).

Modeling Techniques

Modeling techniques that can be used when requirements must be detailed or refined, or in cases in which they address topics not considered during the stakeholder requirements definition and mission analysis include:

- State-charts models (ISO 2011, Section 8.4)
- Scenarios modeling (ISO 2011, Section 6.2.3.1)
- Simulations, prototyping (ISO 2011, Section 6.3.3.2)
- Quality Function Deployment (INCOSE 2011, p. 83)
- Systems Modeling Language (SysML) sequence diagrams, activity diagrams, use cases, state machine diagrams, requirements diagrams (OMG 2010)

- Functional Flow Block Diagram for operational scenarios (Oliver, Kelliher, and Keegan 1997)

Presentation and Quality of Requirements

Generally, requirements are provided in a textual form. Guidelines exist for writing good requirements; they include recommendations about the syntax of requirements statements, wording (exclusions, representation of concepts, etc.), and characteristics (specific, measurable, achievable, feasible, testable, etc.). Refer to (INCOSE 2011, Section 4.2.2.2) and (ISO 2011).

There are several characteristics of both requirements and sets of requirements that are used to aid their development and to verify the implementation of requirements into the solution. Table 3 provides a list and descriptions of the characteristics for individual requirements and Table 4 provides a list and descriptions of characteristics for a set of requirements, as adapted from (ISO 2011, Sections 5.2.5 and 5.2.6).

Table 3. Characteristics of Individual Requirements. (SEBoK Original)

Characteristic	Description
Necessary	The requirement defines an essential capability, characteristic, constraint, and/or quality factor. If it is not included in the set of requirements, a deficiency in capability or characteristic will exist, which cannot be fulfilled by implementing other requirements
Appropriate	The specific intent and amount of detail of the requirement is appropriate to the level of the entity to which it refers (level of abstraction). This includes avoiding unnecessary constraints on the architecture or design to help ensure implementation independence to the extent possible
Unambiguous	The requirement is stated in such a way so that it can be interpreted in only one way
Complete	The requirement sufficiently describes the necessary capability, characteristic, constraint, or quality factor to meet the entity need without needing other information to understand the requirement
Singular	The requirement should state a single capability, characteristic, constraint, or quality factor
Feasible	The requirement can be realized within entity constraints (e.g., cost, schedule, technical, legal, regulatory) with acceptable risk
Verifiable	The requirement is structured and worded such that its realization can be proven (verified) to the customer's satisfaction at the level at which the requirement exists
Correct	The requirement must be an accurate representation of the entity need from which it was transformed
Conforming	The individual requirements should conform to an approved standard template and style for writing requirements, when applicable

Note: Traceability is considered by some sources as a characteristic (ISO 2011). However, a recent viewpoint is that Traceability is actually an attribute of a requirement; that is, something that is appended to the requirement, not an intrinsic characteristic of a requirement (INCOSE 2011). The traceability characteristic or attribute is defined as: The requirement is upwards traceable to specific documented stakeholder statement(s) of need, higher tier requirement, or another source (e.g., a trade or design study). The requirement is also downwards traceable to the specific requirements in the lower tier requirements specifications or other system definition artifacts. That is, all parent-child relationships for the requirement are identified in tracing such that the requirement traces to its source and implementation.

Table 4. Characteristics of a Set of Requirements. (SEBoK Original)

Characteristic	Description

Complete	The requirement set stands alone such that it sufficiently describes the necessary capabilities, characteristics, constraints, and/or quality factors to meet the entity needs without needing other information. In addition, the set does not contain any to be defined (TBD), to be specified (TBS), or to be resolved (TBR) clauses.
Consistent	The set of requirements contains individual requirements that are unique, do not conflict with or overlap with other requirements in the set, and the units and measurement systems they use are homogeneous. The language used within the set of requirements is consistent, i.e., the same word is used throughout the set to mean the same thing.
Feasible	The requirement set can be realized within entity constraints (e.g., cost, schedule, technical, legal, regulatory) with acceptable risk. (Note: Feasible includes the concept of "affordable".)
Comprehensible	The set of requirements must be written such that it is clear as to what is expected by the entity and its relation to the system of which it is a part.
Able to be validated	It must be able to be proven the requirement set will lead to the achievement of the entity needs within the constraints (such as cost, schedule, technical, legal and regulatory compliance).

Requirements in Tables

Requirements may be provided in a table, especially when specifying a set of parameters for the system or a system element. It is good practice to make standard table templates available. For tables, the following conventions apply:

- Invoke each requirements table in the requirements set that clearly points to the table.
- Identify each table with a unique title and table number.
- Include the word “requirements” in the table title.
- Identify the purpose of the table in the text immediately preceding it and include an explanation of how to read and use the table, including context and units.
- For independent-dependent variable situations, organize the table in a way that best accommodates the use of the information.
- Each cell should contain, at most, a single requirement.

Requirements in Flow Charts

Flow charts often contain requirements in a graphical form. These requirements may include logic that must be incorporated into the system, operational requirements, process or procedural requirements, or other situations that are best defined graphically by a sequence of interrelated steps. For flow charts, the following conventions apply:

- Invoke flow charts in the requirements set that clearly points to the flow chart.
- Identify each flow chart with a unique title and figure number.
- Include the word “requirements” in the title of the flow chart.
- Clearly indicate and explain unique symbols that represent requirements in the flow chart.

Requirements in Drawings

Drawings also provide a graphical means to define requirements. The type of requirement defined in a drawing depends on the type of drawing. The following conventions apply:

- Use drawings when they can aid in the description of the following:
  - Spatial Requirements
  - Interface Requirements

    - Layout Requirements
- Invoke drawings in the requirements set that clearly point to the drawing.

Artifacts

This process may create several artifacts, such as:

- System Requirements Document
- System Requirements Justification Document (for traceability purpose)
- System Requirements Database, including traceability, analysis, rationale, decisions, and attributes, where appropriate.
- System External Interface Requirements Document (this document describes the interfaces of the system with external elements of its context of use; the interface requirements can be integrated or not integrated to the system requirements document.

The content, format, layout and ownership of these artifacts will vary depending on who is creating them as well as in which domain they will be utilized. Between them and the outputs of the process, activities should cover the information identified in the first part of this article.

Practical Considerations about System Requirements

There are several **pitfalls** that will inhibit the generation and management of an optimal set of system requirements, as discussed in Table 5.

Table 5. Major Pitfalls with Definition of System Requirements. (SEBoK Original)

Pitfall	Description
Insufficient Analysis of Stakeholder Requirements	If the receivers of the stakeholder requirements do not perform a sufficient critical analysis of them, the consequence could be difficulties translating them into system requirements and the obligation to come back to the stakeholders, losing time.
Insufficient Analysis of Operational Modes and Scenarios	The operational modes and operational scenarios are not sufficiently analyzed or defined by the person in charge of writing the system requirements. Those elements allow the structuring of the system and its use early in the engineering process and help the designer to remember functions and interfaces.
Incomplete Set of System Requirements	If the system requirements are not sufficiently precise and complete, there is a great risk that the design will not have the expected level of quality and that the verification and validation of the system will be delayed.
Lack of Verification Method	Delaying the capture of verification methods and events for each system requirement; identification of the verification approach for each requirement often provides additional insight as to the correctness and necessity of the requirement itself.
Missing traceability	Incorrect or missing traceability of each requirement, both to an upper-level "parent" requirement as well as allocation to an inappropriate system or system element.

The **proven practices** in Table 6 have repeatedly been shown to reduce project risk and cost, foster customer satisfaction, and produce successful system development.

Table 6. Proven Practices for System Requirements. (SEBoK Original)

Practice	Description

Involve Stakeholders	Involve the stakeholders as early as possible in the system requirements development process.
Presence of Rationale	Capture the rationale for each system requirement.
Always Complete before Starting	Check that stakeholder requirements are complete as much as possible before starting the definition of the system requirements.
Peer Reviews	Organize peer reviews of system requirements with applicable subject matter experts.
Modeling Techniques	Use modeling techniques as indicated in sections above.
Requirements Management Tool	Consider using a requirements management tool, especially for more complex projects. This tool should have the capability to trace linkages between system requirements to display relationships. A requirements management tool is intended to facilitate and support the systematic managing of system requirements throughout the project life cycle.
Measures for Requirement Engineering	Use typical measures for requirement engineering; for further information, refer to the *Systems Engineering Leading Indicators Guide* (Roedler et al. 2010). Both process and product measures should be used for requirements engineering. To get the desired insight to facilitate risk-managed requirements engineering, it may be necessary to use more than one measure based on the information needs (risks, objectives, issues) for the requirements. Useful measures include: • Requirements Volatility • Requirements Trends • Requirements Verification Progress (plan vs. actual) • Requirements Validation Progress (plan vs. actual) • TBD and TBR Closure Per Plan • Peer Review Defects

References

Works Cited

Hauser, J. and D. Clausing. 1988. "The House of Quality." *Harvard Business Review.* (May - June 1988).

Hooks, I.F. and K.A. Farry. 2000. *Customer-Centered Products: Creating Successful Products through Smart Requirements Management*. New York, NY, USA: American Management Association.

Hull, M.E.C., K. Jackson, A.J.J. Dick. 2010. *Systems Engineering,* 3rd ed. London, UK: Springer.

INCOSE. 2011. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities,* version 3.2.1. San Diego, CA, USA: International Council on Systems Engineering (INCOSE), INCOSE-TP-2003-002-03.2.1.

ISO/IEC. 2007. *Systems and Software Engineering -- Recommended Practice for Architectural Description of Software-Intensive Systems*. Geneva, Switzerland: International Organization for Standards (ISO)/International Electrotechnical Commission (IEC), ISO/IEC 42010:2007.

ISO/IEC/IEEE. 2011. *Systems and Software Engineering - Requirements Engineering*. Geneva, Switzerland: International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC)/Institute of Electrical and Electronics Engineers (IEEE), ISO/IEC/IEEE 29148.

ISO/IEC/IEEE. 2015.*Systems and Software Engineering - System Life Cycle Processes.* Geneva, Switzerland: International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE). ISO/IEC/IEEE 15288:2015.

Martin, J.N. 1997. *Systems Engineering Guidebook: A Process for Developing Systems and Products,* 1st ed. Boca Raton, FL, USA: CRC Press.

Oliver, D., T. Kelliher, and J. Keegan. 1997. *Engineering Complex Systems with Models and Objects*. New York, NY, USA: McGraw-Hill.

OMG. 2010. *OMG Systems Modeling Language Specification*, version 1.2. Needham, MA, USA: Object Management Group. July 2010.

Primary References

ISO/IEC/IEEE. 2011. *Systems and Software Engineering - Requirements Engineering*. Geneva, Switzerland: International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC)/ Institute of Electrical and Electronics Engineers (IEEE), ISO/IEC/IEEE 29148.

ISO/IEC/IEEE. 2015.*Systems and Software Engineering - System Life Cycle Processes.* Geneva, Switzerland: International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC), Institute of Electrical and Electronics Engineers (IEEE). ISO/IEC/IEEE 15288:2015.

INCOSE. 2015. 'Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities', version 4.0. Hoboken, NJ, USA: John Wiley and Sons, Inc, ISBN: 978-1-118-99940-0.

Lamsweerde, A. van. 2009. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. New York, NY, USA: Wiley.

Additional References

Faisandier, A. 2012. *Systems Opportunities and Requirements*. Belberaud, France: Sinergy'Com.

Hooks, I.F. and K.A. Farry. 2000. *Customer-Centered Products: Creating Successful Products through Smart Requirements Management.* New York, NY, USA: American Management Association.

Hull, M.E.C., K. Jackson, A.J.J. Dick. 2010. *Systems Engineering,* 3rd ed. London, UK: Springer.

Roedler, G., D. Rhodes, C. Jones, and H. Schimmoller. 2010. *Systems Engineering Leading Indicators Guide,* version 2.0. San Diego, CA, USA: International Council on Systems Engineering (INCOSE), INCOSE-TP-2005-001-03.

SEI. 2007. "Requirements Management Process Area" and "Requirements Development Process Area." in *Capability Maturity Model Integrated (CMMI) for Development*, version 1.2. Pittsburgh, PA, USA: Software Engineering Institute (SEI)/Carnegie Mellon University (CMU).

Relevant Videos

- Introduction to Systems Engineering and Requirements

< Previous Article | Parent Article | Next Article >
SEBoK v. 2.4, released 19 May 2021

- 
Powered by
MediaWiki
BKCASE
Powered By
Semantic
MediaWiki