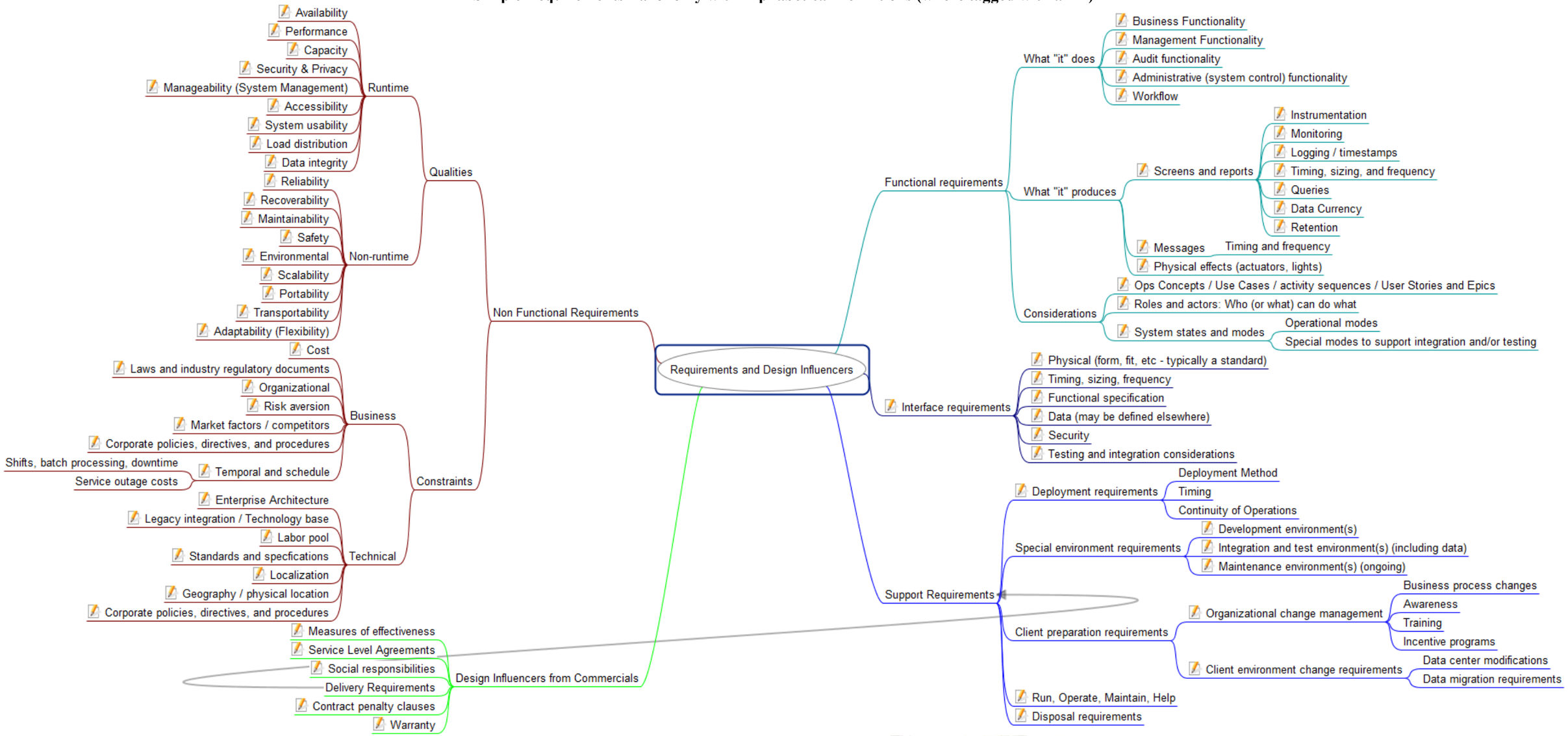
Simple Requirements Taxonomy with Alphabetical Definitions (where tagged with a)

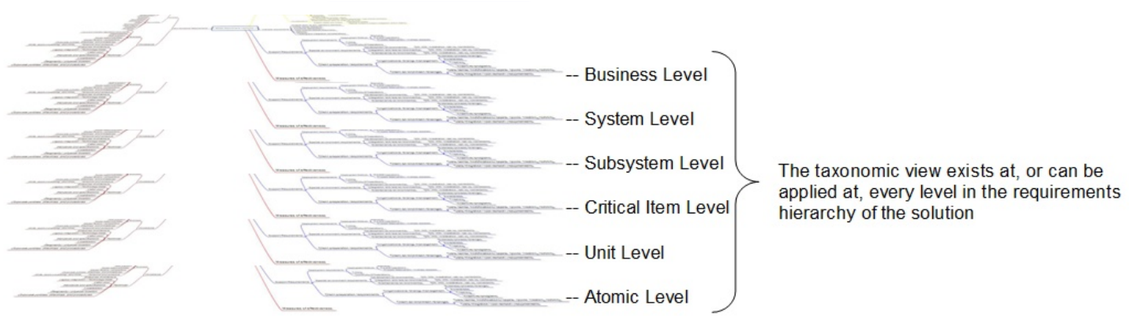


This diagram represents one of three views into the classic requirements space. These views are hierarchical, lifecycle, and taxonomic, defined as follows:

The **hierarchical view** is one that describes the relationship between the requirements at the top level of system, or system of systems, and its composite elements. This hierarchy can be thought of as relating a system to segments, segments to subsystems, subsystems to configuration items, configuration items to components, components to modules, and modules to individual units. Many allow recursive levels (e.g., a subsystem may itself be composed of subsystems), while others have but two levels. The hierarchical view is only looking at the parent/child relationship of requirements and how a requirement may be allocated to elements at the next level of decomposition. This view is relatable to the vertical axis of the classic systems engineering "V" diagram. The taxonomic diagram above can be applied to ANY level of the hierarchy

The second view is the requirement **lifecycle view** - the horizontal axis of the Systems engineering V diagram. This view follows a requirement from left to right across the lifecycle - from creation of the requirement, to review of the requirements, acceptance into the requirements specification, assignment in a test case, definition of successful test results, creation of special test data (if required), creation of a test plan that includes the requirement, and finally the creation and execution of a test script to actually conduct the test.

The diagram above is a third view of requirements - the **taxonomic view.** It represents a taxonomy of requirements types (or areas) that one should consider in any development effort. The requirements in this taxonomy can be found at nearly any hierarchical level of a large or complex system. This list was created as a simple mental checklist to ensure all requirements areas of the system and its components, and associated deployment considerations, have at least been considered and documented as warranted (or not). It expands beyond the basic functional and non-functional areas to include many others that troubled project analysis have found to be necessary but much less commonly addressed in the solution design phase.



Simple Requirements Taxonomy with Alphabetical Definitions (where tagged with a)

Accessibility: Refers to accessibility by the physically disabled. Many countries have regulations that specify both physical (building access) and virtual (screen readers, etc.) accommodations that one must make so that deaf, blind, wheelchair-bound, or otherwise impaired individuals can access the system functionality.

Adaptability (Flexibility): This is also called to as extensibility, and typically refers to the ease of extending the system design to cover additional functionality at a future time. This additional functionality may or may not be planned at the time the solution is designed.

Administrative Functionality: Refers to the controls to be built into the system to modify its operation and behavior as well as more typical activities such as adding or removing users, groups, and permissions. These are also referred to as systems management requirements.

Audit Functionality: Refers to features incorporated into the system to support audits required by business rules and/or government (or industry) regulations. This is typically logging certain transactions or activities within the system.

Availability: typically expressed statistically, these requirements drive fault tolerance and redundancy in the solution. These are frequently driven by an analysis of service outage costs and business risk.

Business Functionality: Not to be confused with business level requirements in a hierarchy, this describes the business functionality of the solution element being specified. Ex: "The Post_Interest module shall calculate and post the daily interest accrued to the account using equation (2) specified below"

Capacity: Capacity requirements include network and interface throughput, memory, storage, and CPU loading.

Client Environment Change Requirements: These requirements address both physical environment and data changes, within client facilities, necessary to support the installation and operation of the solution. Physical space, power, heating, cooling, and data migration (or data conversion) requirements all fall into this category. In some larger deals, an entirely new data center must be developed, the requirements of which should be documented.

Contract Penalty Clauses: May require changes in the design approach to avoid certain eventualities that would require IBM to pay a penalty.

Corporate Policies, directives, and procedures: These can be either business or technical in nature.

Cost: Self explanatory. This constraint may significantly influence both the overall solution and selection of solution components.

Data Currency: This is a subset of data integrity, and refers to how current the data on a screen (or in a report) must be at the time it is displayed.

Data Integrity: Refers to maintaining and assuring the accuracy and consistency of data over its entire life-cycle, to include timeliness of updates and timeliness of availability after an update.

Data Retention: While suggested as a subset of screens and reports, data retention is an issue that spans all data within a system. Data Retention policies will drive the amount of storage and the type of storage (tape, disk, etc).

Deployment Requirements, Continuity of Operations: Specifies any limitation relative to impacting any part of the client's ongoing operations.

Deployment Requirements, Deployment Method: This requirement specifies the deployment to be taken with the system. Common methods include parallel operations, incremental deployment (e.g., upgrading several retail locations each week for an extended period), phased deployment (drops with incremental capability are released on a planned schedule), and big bang. This may be driven by client needs or by the solution developed.

Deployment Requirements, Timing: Refers to any time constraint that may drive deployment. This could be a hard date (as an Olympic system might need), a seasonal date (retail systems should be operational one month prior to major shopping holidays), or time limitations (total duration of the deployment effort from beginning to end - or – time to install new systems in each client site). The time sequence of deployment may also be specified here.

Disposal requirements: These identify needs for the disposal of legacy systems being retired and/or special considerations for the solution being deployed.

Enterprise Architecture constraints: For those clients that maintain and use an overall Enterprise Architecture (EA), that EA will place limitations on the solution in the form of architectural constraints and product selection.

Environmental: In the sense of this taxonomy, environmental requirements specify the environment into which the system is deployed. As such, this tends to focus on temperature and humidity ranges. If this is a mobile application (in the sense of shipboard, airborne, or vehicular), there may be additional specifications on shock and vibration.

Geography / physical location(s): These describe the geographical distribution into which the solution may be deployed.

Instrumentation: Typically driven by screens and reports, these requirements specify what parts of the solution must be instrumented to provide insight from an operations, maintenance, or audit perspective.

Interface Requirements, Data: Contains the definition of the data items that are expected to be exchanged across the interface.

Interface Requirements, Physical: This specifies the form and fit of a physical interface. Today, it is most commonly specified by an open standard containing the physical and signal level information, such as IEEE 1394, USB 3.0, or IEEE 802.3.

Interface Requirements, Security: Contains any data security requirements pertinent to the interface, such as encryption or protocols such as SSL.

Interface Requirements, Test & Integration concerns: Describes any special considerations, such as interface drivers, that may be required to support specific test or integration activities.

Interface Requirements, Timing, sizing, and frequency: This specifies the load characteristics the interface must be capable of supporting. This may be derived through an analysis of data volume and a specified response time at the system level.

Labor Pool: Labor pool refers to either or both of two related areas: available skills of the current labor pool and/or the transfer of the existing labor pool to a new contract holder.

Laws and regulations: Many industries, especially healthcare, automotive, energy and utilities, are tightly regulated by government. Many of these regulations affect the design of client solutions, as they drive safety, security, audit data, and/or privacy aspects of the solution.

Legacy integration / existing technology base: The constraints in this area are typically captured in the Client Environment work product. They describe the existing systems with which, or into which, the solution must be integrated.

Load distribution: This is also known as load balancing and refers to directing incoming requests to other compute resources when a given server is at or near a capacity threshold. These are typically derived requirements driven by performance requirements and the timing and sizing information.

Localization: Localization refers primarily to adapting the solution to local languages and, in some cases, unique laws or regulations that apply in some geographic locations but not others.

Logging / timestamps: Logging defines what values must be captured for potential use at a later time to support debug, maintenance, or auditing. The requirements should also specify the accuracy, and format, of any timestamps associated with the logged values.

Maintainability: These requirements pertain to the solutions ability to detect and isolate defects within the system and facilitate the repair and/or replacement of the defective element in a timely fashion. Sometimes these requirements are specified as a Mean Time to Repair (MTTR). These requirements are not very common in commercial systems, but do commonly appear in industrial and government solutions.

Manageability (System Management): These requirements specify the combination of processes, data, tools, and organization which are needed to manage the solution efficiently and effectively.

Management Functionality: This refers to (generally autonomous) controlling aspects of the execution of the system, such as controlling the timing and execution of certain business functions.

Market factors: Refer to outside, typically competitive, forces that may drive the solution in a direction to counter those forces or to move ahead of the client's competition. These forces may be a primary supporting factor in many architectural decisions.

Measures of Effectiveness: MOE's provide quantifiable benchmarks against which the system concept (initial predictions) and implementation can be compared. These are found more typically in engineering, medical, and military applications.

Messages: Messages are typically created by lower level system elements. Many may be providing information to be incorporated into a higher level screen or report, while other system elements may create messages to accomplish the mission of the system. Warnings and alerts typically fall into the domain of messages.

Monitoring: Defines what data must be monitored, when (time window), how frequently, and to what standard (trending, absolute limits). In some cases, if a monitored value reaches a defined limits, it may results in a message.

Ops Concepts / Use cases / Activity Sequences / User Stories / Epics: These are all tools that can be applied to nearly any level of a hierarchy to provide better insight into functions and operations of the elements of the solution. As a collective set, these are probably best described as use cases, though each of the specific names above is applied to a particular specialization. Operational Concept (aka "ConOps") typically defines the use of capabilities at the highest level (business level). Use cases take those to more detail, and activity sequences lower still. User Stories and Epics achieve the same ends in an agile environment.

Organizational Change Management Requirements: While typically not a technical requirement, it drives overall success of the solution and the IBM-client partnership. Organizational change management requirements typically cover training and awareness programs, business process changes, and incentive programs to motivate organizational change.

Organizational constraints: Refers to a client's organizational separation of duties. This may or may not become a design constraint, in that some transformational initiatives may strive to change organizational constraints to improve overall operations. (Other definitions exist).

Performance: Typically expressed as throughput or response time. One should use a statistical specification rather than absolute.

Physical Effects: This requirement area typically applies only to systems that can control environmental systems (heating, cooling, humidity, lights), physical actuators (robotic systems, security doors), transportation control systems (traffic lights, landing aids), and other systems where the response of the system to certain stimuli requires a response manifested in the physical world.

Portability: The most common use of these requirements is to specify a certain degree of cross-platform compatibility, which typically drives the selection of more open languages and protocols in the solution design.

Queries: This is typically a lower level requirement that specifies what query capability, or what specific queries, the system must support in able to generate the specified screens and reports.

Recoverability: This specifies how quickly o ne must be able to recover operations following an outage due to any cause. It should specify what aspects of operations need to be restored (frequently just a subset) within a given timeframe. There may be several tiers to the recovery process. This requirement typically drives back-up and restore functionality as well as influencing redundancy.

Reliability: This is most frequently expressed as Mean Time Between Failures (MTBF) and is a factor in the overall availability of a system.

Risk Aversion: This refers to a client's preference for taking on vs. avoiding risks. It is typically not documented as a formal constraint, but should be recorded as a consideration when documenting architectural decisions.

Roles and Actors: These define who and what may interact with the solution element and what permission(s) those actors have within that solution element.

Run, Operate, Maintain, and Help: These mainly address delivery requirements in an SO or application maintenance type of engagement. Drives skills / staff levels

Safety: This requirement area is another that will typically cite generally accepted safety standards rather than providing a long list of unique requirements. In some programs, an organization's safety group may be brought in to specify or review these.

Scalability: These requirements specify how the system is able to scale up to handle additional work load. Factors used to specify scalability typically include increase in number of users, number of transactions, network traffic, and size of data managed.

Screens and reports: Screens and reports are typically organized by role, by area of functionality, and/or by use case. These 'outputs' may drive a lower level set of requirements described elsewhere in this definition list.

Security and Privacy: Identify data to be protected, Identify the type of threats to which each type of data is exposed (Accidental corruption or destruction, Deliberate corruption or destruction, Commercial intelligence, Fraud, Hacking, Viruses); Identify threats to physical security (Theft of components, Unauthorized physical access, Physical safety of users), Identify the people who may be the sources of these threats (Data center staff, Other IT staff (for example, development), Non-IT staff of the organization, People outside the organization), Identify any special or unusual security requirements especially with respect to: a. Access to the system, b. Encryption of data, c. Auditability.

Service Level Agreements: These influencers can become major design considerations. These can include everything from system performance and availability to dropped calls and help line responsiveness for help desk operations. An IBM GTS reference can be found by clicking on the link on the right side of this page.

Social Responsibilities: Refers to any of a number of areas of social responsibilities. In the solution design space, this usually devolves to minimal impact on the natural environment, as in responsible disposal of retired solution elements and/or energy efficient data center design.

Special Environment Requirements, Development Environment(s): These requirements should include the specification of any hardware, software, installation, set up, and/or constraints for environment(s) that must be put in place to support the development of the solution.

Special Environment Requirements, Integration and Test Environment(s): These requirements should include the specification of the hardware, software, installation, set up, and constraints for any environment that must be developed to support integration and test of the solution elements, including benchmarking.

Special Environment Requirements, Maintenance Environment(s): These should include the specification of any hardware, software, installation, set up, and constraints for environments that must be developed to support the long term solution maintenance and enhancement activity once the solution has been deployed.

Standards and Specifications: These requirements list the applicable (and reference) standards, and the degree to which they are required vs. referenced as guidance.

System States and Modes: These requirements typically appear only at the system or subsystem level within a hierarchy. States and modes refer to operational conditions in which the system may be placed – or may find itself through certain natural events. Examples of states include operational, offline (machines are turned off or the applications are not running), disconnected (machines are up and running, but there is no network connectivity perhaps due to natural disaster), and underway (for vehicular systems). Modes are frequently considered to exist within a given state and refer to a particular act or way of accomplishing work. Batch processing may be a mode distinct from real time transaction processing, for example.

System Usability: These document any 'ease of use' requirements. For testability purposes, these are best documented by citing a standard to which the system must adhere (AIAA HCI standards, IBM standards, or any standards the client may have). Please note that "The system shall be easy to use" is not testable.

Temporal and Schedule Constraints: These requirements typically identify shifts or time periods when certain types of processing (batch) may be performed, or when the system may be taken down for maintenance.

Timing, Sizing, and frequency: This refers to system outputs such as screens and reports. Timing refers to when the screens or reports are to be generated,. Frequency specifies how frequently the screen is updated (or a report produced). Sizing refers to the amount of data contained within the report (to facilitate overall system sizing).

Transportability: Typically used to specify transportation requirement for moving a system from one location to another. This is commonly used in man-packable systems or portable data center implementations.

Warranty: Warranty will affect your pricing in that the pricing needs to cover maintenance of defects found during the warranty period. This is based on your expected defect rate, which is affected by your overall development approach and environment.

Workflow Functionality: Refers to the automation aspects of a sequential business process being implemented by the system.