ICT Project Guidance

Definition -   
System Quality-Based   
Non-Functional Requirements Development

Version:

0.2

Author:

Sky Sigal, Solution Architect

## Purpose

The purpose of this document is to provide guidance to inform the development of valuable quality requirements of systems, as the basis for Requests for Proposals and Contractual agreements between consumers and suppliers.

## Synopsis

Non-Functional Requirements (NFRs) is a deprecated term based on an analysis of outcomes versus expectations. The international Institute of Business Analysts (IBBA)’s current version of the Business Analysis Body of Knowledge (BABOK) recommends to instead developing System Quality Requirements guided by ISO-25010.

This document summarises the process to do this, taking also into account this organisation’ market & sector constraints, and organisation’s current desires: Quality Requirements are organised in a tiered, pyramidal manner to diminish the effort of RFx respondents to determine project specifics versus default quality expectations.

## Contents

[Purpose 1](#_Toc156143647)

[Synopsis 1](#_Toc156143648)

[Contents 2](#_Toc156143649)

[Background 4](#_Toc156143650)

[Context 4](#_Toc156143651)

[Known Issues 5](#_Toc156143652)

[ISO Defined Qualities 5](#_Toc156143653)

[Organisation 5](#_Toc156143654)

[Hierarchical Tiers 6](#_Toc156143655)

[Qualities 7](#_Toc156143656)

[Exemptions 8](#_Toc156143657)

[Requirement Attributes 8](#_Toc156143658)

[Custom Development Quality Requirements 8](#_Toc156143659)

[Appendices 10](#_Toc156143660)

[Appendix A - Document Information 10](#_Toc156143661)

[Images 10](#_Toc156143662)

[Tables 10](#_Toc156143663)

[References 10](#_Toc156143664)

[Review Distribution 10](#_Toc156143665)

[Audience 10](#_Toc156143666)

[Diagrams 10](#_Toc156143667)

[Terms 11](#_Toc156143668)

[Appendices B – ISO-25010 Qualities 11](#_Toc156143669)

[Functional Suitability 11](#_Toc156143670)

[Functional Completeness 12](#_Toc156143671)

[Functional Correctness 12](#_Toc156143672)

[Functional Appropriateness 12](#_Toc156143673)

[Performance Efficiency 12](#_Toc156143674)

[Time behaviour 12](#_Toc156143675)

[Resource Utilisation 12](#_Toc156143676)

[Capacity 12](#_Toc156143677)

[Compatibility 12](#_Toc156143678)

[Co-existence 12](#_Toc156143679)

[Interoperability 12](#_Toc156143680)

[Usability 13](#_Toc156143681)

[Appropriateness recognisability 13](#_Toc156143682)

[Learnability 13](#_Toc156143683)

[Operability 13](#_Toc156143684)

[User Error Protection 13](#_Toc156143685)

[User Interface Aesthetics 13](#_Toc156143686)

[Accessibility 13](#_Toc156143687)

[Reliability 13](#_Toc156143688)

[Maturity 14](#_Toc156143689)

[Availability 14](#_Toc156143690)

[Fault tolerance 14](#_Toc156143691)

[Recoverability 14](#_Toc156143692)

[Security 14](#_Toc156143693)

[Confidentiality 14](#_Toc156143694)

[Integrity 14](#_Toc156143695)

[Non-Repudiation (of Action) and Accountability (by Whom) 14](#_Toc156143696)

[Authenticity 14](#_Toc156143697)

[Availability 14](#_Toc156143698)

[Maintainability 15](#_Toc156143699)

[Modularity 15](#_Toc156143700)

[Reusability 15](#_Toc156143701)

[Analysability 15](#_Toc156143702)

[Modifiability 15](#_Toc156143703)

[Testability 15](#_Toc156143704)

[Portability 15](#_Toc156143705)

[Adaptability 15](#_Toc156143706)

[Installability 15](#_Toc156143707)

[Replaceability 15](#_Toc156143708)

[Deliverability 16](#_Toc156143709)

[Regulation 16](#_Toc156143710)

[Privacy 16](#_Toc156143711)

[Data Sovereignty 16](#_Toc156143712)

[Appendix C – Attribute Qualities 16](#_Toc156143713)

[Requirement Attributes 16](#_Toc156143714)

[ID 16](#_Toc156143715)

[[Diagram] Title 16](#_Toc156143716)

[Statement 16](#_Toc156143717)

[Fit/Acceptance Criteria 17](#_Toc156143718)

[Details 17](#_Toc156143719)

[Questions, Response and Analysis 17](#_Toc156143720)

[Metadata 17](#_Toc156143721)

## Background

Requirements are contractual in nature[[1]](#footnote-2), used to formalise agreement as to capabilities or conditions between supplier and consumer before commitment.

## Context

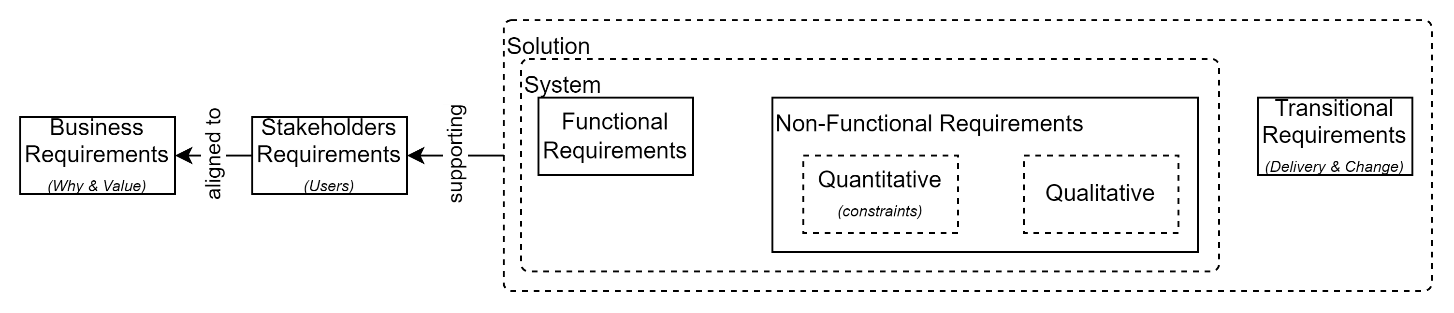


Figure : BABOK defined types of Requirements.

Non-Functional Quality Requirements are one of the 4 key types of BABOK defined Requirements:

* **Business Requirements (BR)**, defining the sponsors expected business outcomes and service value,
* **Stakeholder Requirements (SR)**, defining what stakeholders and system end uses require, to deliver the above defined business objectives,
* **Solution Requirements**, the logical set[[2]](#footnote-3) of:
  + **System Requirements**, the logical set of:
    - **Functional Requirements (FR)**, defining the requirements of users and stakeholders to operate the system to deliver the expected business outcomes,
    - **Non-Functional Requirements (NFR)**, defining the qualities of the system as *qualitative* statements referring to *quantitative* constraints,
  + **Transitional Requirements** (TR), defining requirements to transition from current to operational state.

## Known Issues

The lack of specificity of the term “Non-Functional” have traditionally caused undesirable outcomes, with NFRs becoming a disorganized "parking ground" for requirements that are not easily identifiable as Business User Functional Requirements. NFRs often contain both functional requirements for Support, Operations, Maintenance Deployment, Quality Assurance and Security Specialists, and project delivery procedure requirements[[3]](#footnote-4).

Additionally, the reviewing of Requirements to develop formal responses to RFxs is time consuming and costly, leading to a desire to reduce the quantity of Requirements in an RFx to increase the likelihood of attracting respondents.

Common outcomes from the above were critical omissions, leading unplanned-for costs, and many times, project failure to delivery to expectations.

## ISO Defined Qualities

To address the first issue, the latest version[[4]](#footnote-5) of International Institute of Business Analysis (IIBA)’s Business Analysts Book of Knowledge (BABOK) recommends developing “Quality Requirements”, according to the following ISO guidelines:

* ISO-25010[[5]](#footnote-6) defines valued qualities of *systems*,
* ISO-25012 defines valued qualities of *data* managed *within* systems, and
* ISO-25022 defines valued qualities of the *experience* users haveusing the system to retrieve, use and manage the data within it.

Note:  
To clarify outcomes many organisations have dropped the use of the term “NFRs” in preference for Quality Requirements or Quality Attributes. We endorse and recommend this change.

## Organisation

TO address the second issue, BABOK does *not* recommend an arbitrary approach to trimming requirements.

Instead, Non-Functional Requirements should be complete to minimise miscomprehensions as to expectations, failure to deliver and potential litigation.

However organisation of the requirements can be employed to reduce the amount of time and effort to review and comprehend the expectations.

Our recommendation is to organise and present requirements as a set of hierarchical documents, as described below.

### Hierarchical Tiers

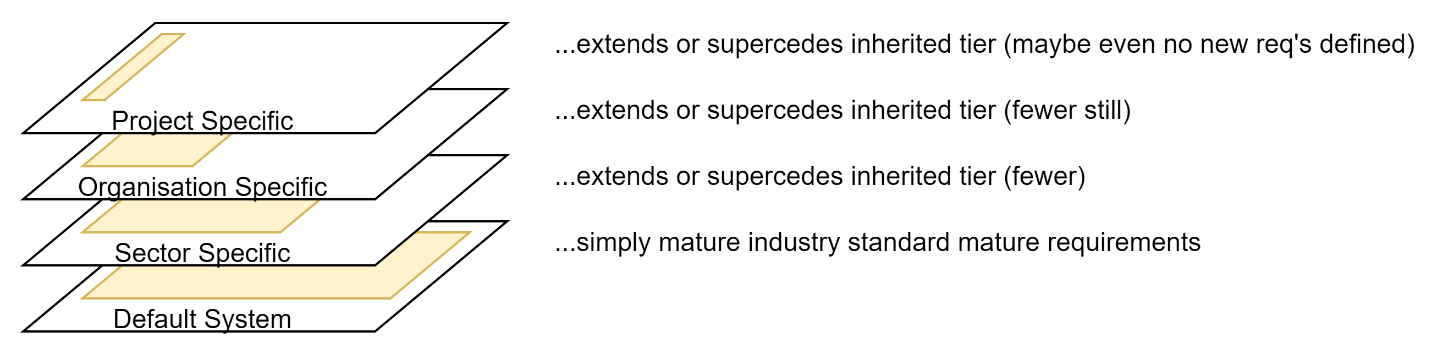


Figure : Hierarchically Tiered Requirements

Requirements are organised according to tiered Scopes -- each tier extending or superseding one or more requirements inherited from a more general tier.

**Default System Requirements**

At the bottom are Default system requirements, suitable for the largest number of contexts or objectives, with the largest number of requirements., irrespective of business domain or organisation.

Important:  
While the Default Tier is the one with the largest number of requirements, requiring the largest investment of time and expense to review once, the requirements are industry standard, simply describing best practice, so should be absolutely no surprise to mature software service delivery firms.

**Scope-Specific Requirements**

Inherited from that are *Scope-specific* requirements. The number of requirements is substantially fewer than in the Default system tier – usually only a handful to address country or business sector specific legislation or other agreements.

**Organisation-Specific Requirements**

Inheriting from Sector specific requirements are *Organisation-specific* standard requirements, addressing the two types of service deployments they routinely purchase: \*

* **Managed**: ones they host, Maintain, Operate and Support themselves (as PaaS or IaaS based systems), and
* **Subscribed**: ones they subscribe to and are hosted elsewhere (following the SaaS procurement model) and Operate and Support.

The organisation specific requirements are usually very small, having mostly all been addressed in the default and sector specific tier, and usually are just outlining some implementation and processes specifics.

All of these lower tiers of requirements (Default/General, Sector-specific, Organisation-specific) groups rarely change, so once a respondent has familiarised themselves with them, they need only focus on any Project specific requirements.

The number of Project specific NFRs – as they neither include Functional or Transitional -- requirements should, in theory, be zero, but certainly will be only a small handful.

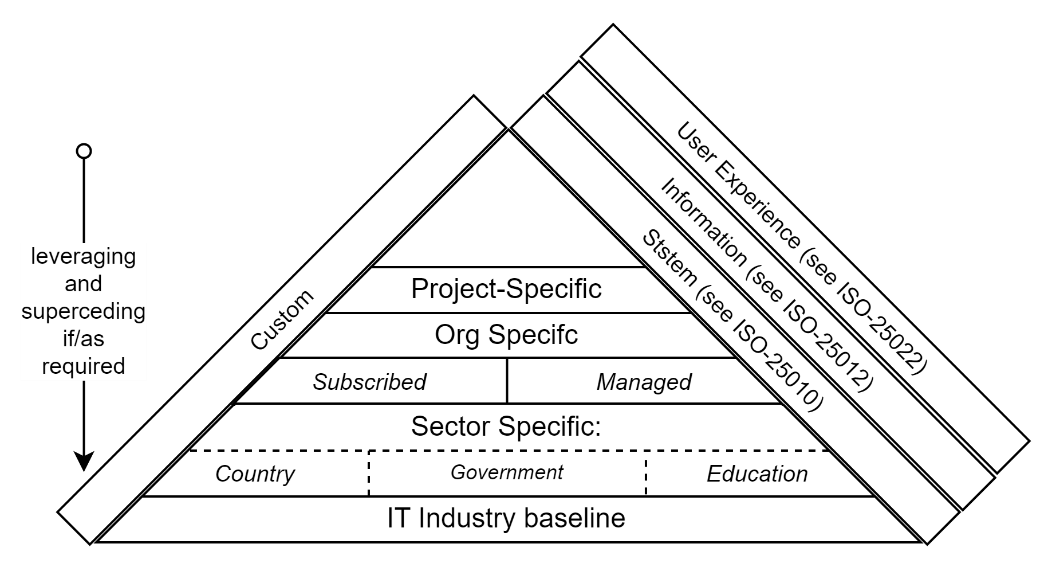


Figure : Pyramid of Context Qualities

**The above organisation as stacked dependent tiers, managed in separate documents, is purposeful:** by using separate managed documents, the rate of change to requirements in lower tiers is limited, providing a solid base of considered organised requirements that can be reused project to project, reducing the risk of introducing novel approaches to previously solved IT issues and/or omitting important requirements.

### Qualities

Within each Tier of requirements, requirements are organised according to the recommended ISO standards-based headers, adding current headers[[6]](#footnote-7).



## Exemptions

Not all solutions can reasonably meet all requirements that are marked as MUSTs.   
For example, proposed solutions may rely on the use of 3rd party SaaS service providers -- whom it is improbable will agree to modifying their service to meet project specific requirements. In such cases, it is up to the presenter to submit a work around that can meet the same obligations and/or objectives.

## Requirement Attributes

Requirements are expected to have a minimum number of attributes (Title, Statement, etc.).

These are covered in the Appendices.

## Requirement of Requirements

The Appendices contains a section containing Requirements to follow when developing Requirements.

## Custom Development Quality Requirements

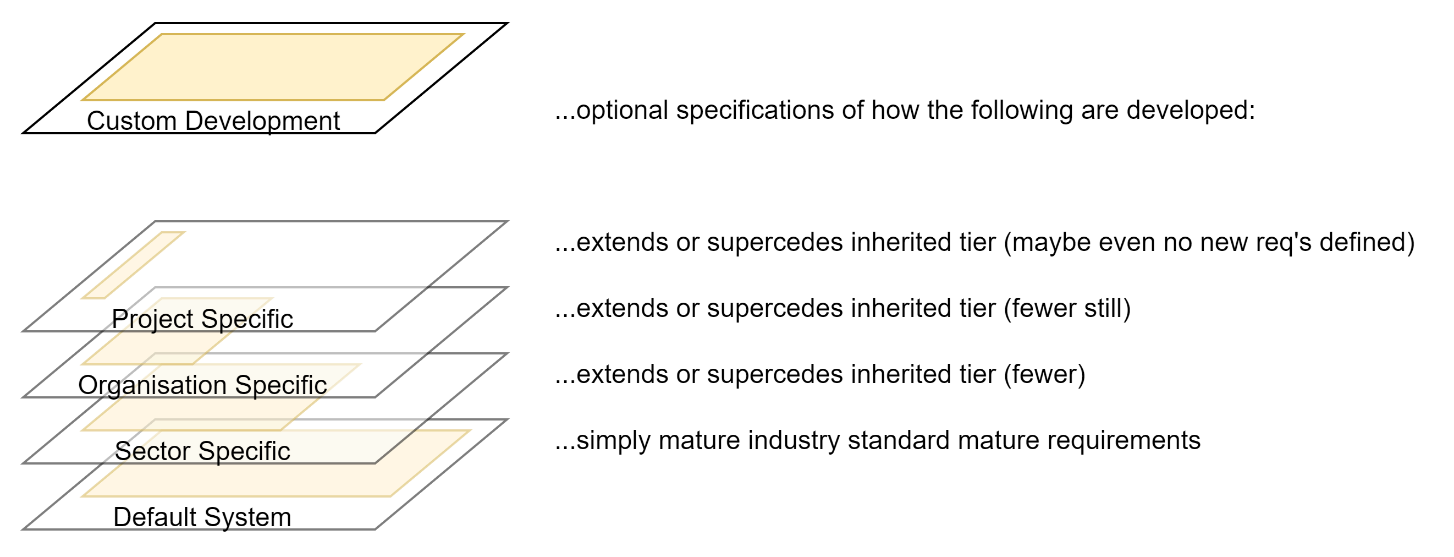


Figure : Custom Development Quality Requirements

The requirements captured in Default, Sector, Organisation and Project specific Quality requirements define *what* qualities are expected. They attempt to avoid as much as reasonable possible defining *how* they are expected to be delivered. This is so RFx respondents can respond with a solution that relies on either a custom developed system, a service or product, or a mix thereof.

While solutions that are purchased or subscribed to are not qualified to ensure they are maintainable, modular, and improvable at a code level, custom developed systems are held to a higher standard and are.

Hence -- in addition to the general quality requirements defined in this document -- custom developed systems must meet the obligations listed in the supplemental *Project Guidance – Definition – Custom Developed Solution Quality Requirements*.

## Summary

The development of systems that have qualities that stakeholders value begins by defining quality statements and contracting for the delivery of services that meet them. This document provides guidance on how to organise this outcome to contribute to de-risking project delivery.

Appendices

Appendix A - Document Information

Versions

* 1. Initial Draft
  2. Addition of Appendices

### Images

[Figure 1: BABOK defined types of Requirements. 6](#_Toc156143515)

[Figure 2: Hierarchically Tiered Requirements 8](#_Toc156143516)

[Figure 3: Pyramid of Context Qualities 9](#_Toc156143517)

[Figure 4: Custom Development Quality Requirements 10](#_Toc156143518)

### Tables

### References

*ICT Project Guidance – Requirement Development*

*Project Guidance – Definition – Custom Developed Solution Quality Requirements*

**There are no sources in the current document.**

### Review Distribution

The document was distributed for review as below:

|  |  |
| --- | --- |
| Identity | Notes |
|  |  |
|  |  |
|  |  |

### Audience

The document is technical in nature, but parts are expected to be read and/or validated by a non-technical audience.

### Diagrams

Diagrams are developed for a wide audience. Unless specifically for a technical audience, where the use of industry standard diagram types (ArchiMate, UML, C4), is appropriate, diagrams are developed as simple “box & line” monochrome diagrams.

### Terms

Refer to the project’s Glossary.

##### Proprietary Code

: code that indirectly or directly belongs to and is controlled by a vendor distinct from the organisation purchasing the development of the solution (the Ministry). Does not include code used to develop a SaaS.

##### ‘Custom Code’

: all information required to develop, deploy, configure, integrate, provision any aspect of the service, excluding proprietary information such as Software as a Service (SaaS).

Custom System Code: Custom code of the System itself.

##### Custom Deliverables

: Code & Documentation.

Prod Data Environments: environments that contain production data that includes confidential information.

Non-Prod Data Environment: *all* other environments (BT, DT, ST, UT, PP, TR, etc.) than the Prod environment.

##### Organisation Provided

: if the organisation that is procuring the service has a service, then it must be used, unless mutually decided otherwise. If the organisation does not have the service, governance can mutually agree to use an alternate service until the organisation provides one to which it can be moved.

##### SaaS

: *Software as a Service*, a managed service rented via a subscription account.

Appendix B – FAQ

### What is the difference between Requirements versus User Stories

Each has a different specific purpose: Requirements are Contractual artefacts - User Stories are work item management artefacts.

Requirements are used to outline Obligations, Recommendations and Prohibitions *prior* to contracts being agreed upon and work commencing, and User Stories are developed after agreement has been reached, to track work that aligns with them.

If Requirements are complete, the User Stories define How the Requirements are to be delivered. If Requirements are not complete, User Stories are used to develop How the delivery of new outcomes are to be delivered.

Both have a distinct format. For example, whereas a User Story often follows a "As a  [StakeholderType], I Desire [Outcome] for [Benefit|Value]" format, and MUST have multiple specific Acceptance Tests, the structural parts of a Requirement are outlined in another Appendix.

Note:  
User Stories that are not traceable to Requirements are valuable during Postmortems to add requirements to default requirements to improve delivery of subsequent projects.

- Appendix B - ISO-25010 Qualities

ISO-25010 defines Headers and sub headers, each with their individual descriptions.

These are as listed below.

### Functional Suitability

This degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions.

This quality is composed of the sub-qualities listed below.

#### Functional Completeness

: the Degree to which the set of functions covers all the specified tasks and user objectives.

#### Functional Correctness

: the Degree to which a product or system provides the correct results with the needed degree of precision.

#### Functional Appropriateness

: the Degree to which the functions facilitate the accomplishment of specified tasks and objectives.

### Performance Efficiency

The degree of the performance relative to the amount of resources used under stated conditions.

Efficiency, throughput, capacity, response time and resource consumption for Online Transaction Processing (OLTP) and background processes.

The system's ability to complete defined volumes of computer assisted Business Transactions (online and background operations) within acceptable times and with an acceptable consumption of resource.

This quality is composed of the following sub-qualities:

#### Time behaviour

: is the degree to which the response and processing times and throughput rates of a solution, when performing its functions, meets requirements.

#### Resource Utilisation

: is the degree to which the amounts and types of resources used by a product or system, when performing its functions, meets requirements.

#### Capacity

: is the degree to which the maximum limits of the solution meet or exceed requirements.

### Compatibility

Degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions while sharing the same hardware or software environment.

This quality is composed of the sub-qualities listed below.

#### Co-existence

: the Degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.

#### Interoperability

: the Degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.

### Usability

The Degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

Usability is how easy and efficient it is for an end user to use the solution correctly, accurately, and safely.

It also ensures constancy with other organisation applications.

It is sometimes called system ergonomics, accessibility and ease of use.

This quality is composed of the sub-qualities listed below.

#### Appropriateness recognisability

: is the degree to which users can recognize whether a solution is appropriate for their needs.

#### Learnability

: is the degree to which a solution enables the user to learn how to use it with effectiveness, efficiency and in the case of an emergency.

#### Operability

is the degree to which a product is easy to operate, control and appropriate to use.

#### User Error Protection

: is the degree to which a solution protects users against making errors.

#### User Interface Aesthetics

: is the degree to which a user interface enables pleasing and satisfying interaction for the user.

#### Accessibility

: is the degree to which a solution can be used by people with the widest range of characteristics and capabilities to achieve a specific goal in a specified context of use.

### Reliability

The Degree to which a system, product or component performs specified functions under specified conditions for a specified period of time.

Reliability of a solution is - a request being processed by that system according to agreed business logic and a valid response being returned by the system in the time expected. The time expected is influenced and constrained by the performance requirements of the system.

Fault tolerance is defined as the immunity of the system in the event of an unexpected internal or external fault. This is evaluated in the context of the likelihood of the fault, the severity of the fault and resulting impact on end users.

*Recoverability* is a subset of *Reliability* andis the ability to re-establish the normal functioning of a system after a significant external event, such as a disaster. A disaster implies total loss of access or use of the system by end users (e.g., production) where the outage has, or will, extend beyond the Maximum Tolerable Downtime (MTD).

The MTD and ‘normal functionality’ are agreed between IT (or the vendor) and the business owner of the system in the Service Level Agreement (SLA).

*Recoverability* uses activities such as data protection, Disaster Recovery (DR), and Business Continuity Planning (BCP) to ensure the recoverability of systems

This quality is composed of the sub-qualities listed below.

#### Maturity

: is the degree to which a solution needs for reliability, under normal operation.

#### Availability

: is the degree to which a solution is operational and accessible when required for use.

#### Fault tolerance

: is the degree to which a solution operates as intended despite the presence of hardware, software or user faults.

#### Recoverability

: is the degree to which, in the event of an interruption or failure, a solution can recover the data directly affected and re-establish the desired system state.

### Security

The degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization.

The practice of maintaining the confidentiality, privacy, integrity and accountability of data changes by controlling authorised access, use and disclosure, while preventing unauthorised use, disruption, modification or destruction.

ISO-25010 recommends considering the following qualities and their descriptions when considering a solution:

#### Confidentiality

: the degree to which the solution ensures data is accessible only by those authorised to do so.

#### Integrity

: the degree to which the solution prevents unauthorised access, modification of systems and the information they manage.

#### Non-Repudiation (of Action) and Accountability (by Whom)

: are associated in that non-repudiation defines the degree to which the solution can prove that actions have been taken, and accountability is being able to associate the non-repudiable (audited) activity to a specific user.

#### Authenticity

: defines the degree to which the identity of a user can be claimed.

#### Availability

: a prerequisite for Security but is treated separately, under Reliability.

Note:  
Security and Privacy risk assessments, and matching Statement of Applicability listing required controls for the solution will be conducted during the solution’s design phase.  That process will further augment the security requirements listed below.

### Maintainability

The degree of effectiveness and efficiency with which a product or system can be modified to improve it, correct it or adapt it to changes in environment, and in requirements.

Maintainability is the ability to maintain the system efficiently: find and remove faults, improve performance, carry out modifications and infrastructure upgrades. It is also known as modify-ability, enhancement, fault detection, isolation and repair.

Traceability is the ability to retain details of specified business, transactional, and system activity and data changes, including normal, abnormal, and error conditions with selective control. It is also called auditability and audit trail.

This quality is composed of the sub-qualities listed below.

#### Modularity

: is the degree to which a solution is composed of discrete components such that a change to one has minimal impact on others.

#### Reusability

: is the degree to which an asset can be used in more than one system or reused to build other assets.

#### Analysability

: is the degree of effectiveness and efficiency with which it is possible to assess the impact of a solution, a change, a failure, or determine what requires change.

#### Modifiability

: is the degree to which a solution can be effectively and efficiently modified without introducing defects or degrading operational quality.

#### Testability

: is the degree to which test criteria can be established, and determination of whether the solution meets them.

### Portability

The Degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another.

This quality is composed of the sub-qualities listed below.

#### Adaptability

: is the degree to which a solution can effectively and efficiently be adapted for different and evolving hardware, software, or environments.

#### Installability

: is the degree of effectiveness and efficiency in which a solution can be successfully installed/uninstalled in a specified environment.

#### Replaceability

: is the degree to which a solution can replace another system for the same purpose in the same environment.

### Deliverability

While *Installability* is a quality defined under *ISO-25010/Portability* (see further down), the subject of Delivery wider, and called out first, with its own section.

### Regulation

Regulation is not a defined ISO-25010 concern, but since the last edition, has become a first-class quality

#### Privacy

The privacy of users is now paramount.

#### Data Sovereignty

the locality of where data is persisted has become a regional and international concern.

## Appendix C - Requirement Attributes

The following are the recommended Attributes of Functional and Non Functional Requirements.

### ID

Each requirement has a unique ID for unambiguous referencing later, from contracts and work items.

As is customary, the Quality Requirement starts with QR- before expressing succinctly what Tier and Category they belong to: QR-{Tier}-{Quality}-{number}.

### [Diagram] Title

Each requirement is provided a short title. The title is intended to be unique, while being both less opaque than an ID, and short enough that they could be used within diagrams.

### Statement

A Requirement Statement is expected to be succinct, define its prioritisation, and developed by following SMART guidance.

Good practice dictates that requirements are:

* SMART[[7]](#footnote-8) (**S**pecific, **M**easurable, (pragmatically) **A**chievable, **R**elevant, **T**imely) statements, prioritised as MUST, SHOULD, MUST NOT, SHOULD NOT[[8]](#footnote-9),
* developed in a CLEAR manner (**C**ollaboratively, **L**imited scope, **E**valuated, **A**ppropriate, **R**esource Conscience) manner,

### Fit/Acceptance Criteria

Acceptance Criteria, sometimes referred to as Fit statements, define specific tests the of the outcome.

### Details

Requirement statements are expected to be terse, therefore Implementation and Outcomes may require further description.

### Questions, Response and Analysis

RFx Respondents should review the Requirements Statement, Fit/Acceptance Criteria, Details and any guiding Questions, to succinctly respond in the Response column as to how their proposed solution meets the intended outcomes.

Their response is analysed and notes are recorded as to whether the proposed solution sufficiently aligns with the requirements statement and its Fit/Acceptance Criteria.

### Metadata

A column for recording tags, or references to previous or other requirements is recommended.

Appendix D – Requirements of Requirements

It seems only fitting to use Requirements to define Requirements for Requirements.

Table : Requirements of Requirements

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | # | ID | Statement | Rationale & Details |
|  | RR-01 | Default/ Requirements/ When | MUST NOT develop SoWs or other Contracts that are not bound to Requirements. | It is irrational to develop agreements without constraints defined. |
|  | RR-02 | Default/ Requirements/ UserStories | MUST NOT use User Stories as the basis for Contracts. | Requirements are Contractual obligations, defined and agreed upon before Contracts and Agreements are made, and work commences.  User Stories are work item definition and prioritisation artefacts, developed after contracts are signed and work is in process. |
|  | RR-03 | Default/ Requirements/ Wording/ SHOULD | SHOULD avoid the use of SHOULD in contractual requirements - they lead to ambiguity as to whether they will be delivered. | MUST is an obligation. SHOULD is only a recommendation with no contractual obligation.  Instead of using SHOULD, let the RFx Respondent respond they will not meet the requirement, propose an alternate approach that they agree to implement. |
|  | RR-04 | Default/ Requirements/ Wording/ SHALL | MUST NOT use "SHALL" - use the imperative MUST. | Reduce litigation: See: <https://www.plainlanguage.gov/guidelines/conversational/shall-and-must/> |
|  | RR-05 | Default/ Requirements/ Wording/ WANT | MUST NOT use "WANT/DESIRE" for the same reasons. | It's one of the reasons why User Stories are inappropriate for the use in Contract. |
|  | RR-06 | Default/ Requirements/ Outcomes | SHOULD Describe the Outcome desired and the level to which it is to be delivered, while avoiding describing How it is achieved (to in turn permit the maximum design and cost flexibility by implementors). | The outcome is increased value and/or reduction of risk.  The output is an actual artefact or product, which constrains how RFx respondents can implement the solution. |
|  | RR-07 | Default/ Requirements/ Criteria | Whereas projects should expect to develop full Business Requirements, they SHOULD expect to only have to add Criteria metrics (Throughput, Storage amount, etc.) to Quality Requirements. | If requirements are developed intelligently, the Quality Requirements will use statements that refer back to a short list of Quantities that is reviewed and/or updated per project.  Don’t embed Quantities into Statements as they are easy to miss when preparing the Requirements. |
|  | RR-08 | Default/ Requirements/ Eliciting/ Stakeholders | When developing Business and User Requirements, ensure the SMEs of all Stakeholder groups are engaged - not just Business Service Providers.  Consider finding distinct SMEs for the list below -- Service Consumers, Service Providers, Cultural Representative Specialists, Support Specialists, Operations Specialists, Maintenance Specialists, Assurance (Security, Privacy, Performance, Quality) Specialists. | Stakeholder Requirements are not fit for purpose if they only capture Business Service Provider SME’s perspective, acting as a Service Consumer SME.  It is important to get SMEs for all groups of users who will actually be using the service. |
|  | RR-09 | Default/ Requirements/ Postmortems | SHOULD include a requirement to perform a Post-Mortem to improve the Enterprise Requirements. | Start from the beginning, looking for UserStories that had to be added due to missing Requirements. |
|  | RR-10 | Default/ Requirements/ Wording/ All | To avoid ambiguity MUST not use the word "all". | "Deliver all documentation" will mean different things to different stakeholders. |
|  | RR-11 | Default/ Requirements/ Wording/ IF | You MAY use Conditional Statements (e.g.: "IF hosted as an Service (SaaS)...") to permit general Requirements being excluded/included under certain conditions. | Very useful to exclude the requirement if not required.  Examples: - “IF developing custom systems,…” - “IF hosted and managed by the organisation,…” |
|  | RR-12 | Default/ Requirements/ Improvements | Requirements MUST be reviewed regularly to look for whether outcomes are still achievable, and not introducing technical How statements. | Continuously improve requirements. |

1. as opposed to User Stories or other forms of Work Items used to define and priorities effort *after* the contractual phase, during subsequent development and delivery phases. [↑](#footnote-ref-2)
2. Defining something as the ‘logical set’ means there is no specific document with that title -- it just provides a single term to conveniently refer to all of the constituent parts of the set. [↑](#footnote-ref-3)
3. Better referred to as Transitional Requirements (TRs). [↑](#footnote-ref-4)
4. V3, released in April 2015. [↑](#footnote-ref-5)
5. Published 2011. [↑](#footnote-ref-6)
6. ISO-25010 was developed over a decade ago, prior to the rise in importance of Privacy, Deliverability by automation, and meeting Regulatory concerns. These headers are added to these documents. [↑](#footnote-ref-7)
7. BABOK proposes that requirements have: “Atomicity, Completeness, Consistency, Conciseness, Feasibility, Unambiguity, Testability, Prioritization and Understandability” [↑](#footnote-ref-8)
8. [↑](#footnote-ref-9)