DRAFT - ICT Project Guidance

<TODO>

Version:

0.1

Author:

<name>, <role>, <group>

## Purpose

The purpose of this document is to diminish risk by identifying and agreeing on expectations on how to make the delivered solution’s systems(s) more easily meet user experience expectations[[1]](#footnote-2), while the system is secure and more easily operated, monitored, and maintained[[2]](#footnote-3).

## Synopsis

**Technical Requirements** apply to **custom code** – specifically **custom system code**.

## Contents

[Purpose 1](#_Toc159220233)

[Synopsis 1](#_Toc159220234)

[Contents 2](#_Toc159220235)

[Background 3](#_Toc159220236)

[Outcomes [Objectives] 3](#_Toc159220237)

[Options [Considered & Selected] 3](#_Toc159220238)

[Constraints 3](#_Toc159220239)

[Assumptions 3](#_Toc159220240)

[Dependencies 3](#_Toc159220241)

[Decisions 3](#_Toc159220242)

[Deliverables/Outputs 3](#_Toc159220243)

[Heading Level 3 3](#_Toc159220244)

[Heading Level 4 3](#_Toc159220245)

[Appendices 5](#_Toc159220246)

[Appendix A - Document Information 5](#_Toc159220247)

[Versions 5](#_Toc159220248)

[Images 5](#_Toc159220249)

[Tables 5](#_Toc159220250)

[References 5](#_Toc159220251)

[Review Distribution 5](#_Toc159220252)

[Audience 5](#_Toc159220253)

[Structure 5](#_Toc159220254)

[Diagrams 6](#_Toc159220255)

[Terms 6](#_Toc159220256)

[Appendix B - FAQs 6](#_Toc159220257)

[Appendix C – Requirement Record Template 6](#_Toc159220258)

## Background

…

## Outcomes [Objectives]

…

# Technical Requirements

The following **requirements** are applicable to **custom developed** **SaaP** systems and **custom supporting code**.

## System Design

##### TR-ID: **Service System Design**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | **Service** **system component** design **MUST** align with **Domain Driven Design assembly separation** guidance**.** |
| **Rationale** | **Domain Driven Design (DDD)** improves maintainability by improving discoverability, recognisability , analysability of code within complex systems. |
| **Details** | Any system intended to meet the requirements for a large enterprise, country or public use case will quickly become “complex”.  A layout similar to the following is expected:   * LogicalModule.Presentation.Web (for web services) * LogicalModule.Architecture (an orchestration layer, of both) * LogicalModule.Domain (the business logic, kept separate from) * LogicalModule.Infrastructure (the technical services) * LogicalModule.Shared (common contracts, messages & entities) |
| **Prompts** | … |

##### TR-ID: **Tiers**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | Use separate Tiers for Data than for Logic |
| **Rationale** | … |
| **Details** | Avoid using domain logic within the data **tier** (i.e., avoid using **Stored Procedures** for domain logic). |
| **Prompts** | … |

##### TR-ID: **Application Logic Tier**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | Avoid using domain logic within the data **tier** (i.e., avoid using **Stored Procedures** for domain logic). |
| **Prompts** | … |

### Design

##### TR-ID: **Tenancies**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | Resources MUST be describable with metadata. |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

##### TR-ID: **Metadata**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | Resources MUST be describable with metadata. |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

##### TR-ID: **Grouping**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

## Development

### General

The following are requirements that pertain in general to development, whether it be for service, service client, supporting custom code.

##### TR-ID: **Technology Choices**

|  |  |
| --- | --- |
| **Category** | Technical/Development |
| **Statement** | Technologies used to design, develop, deliver custom solutions MUST use technologies acceptable by the sponsor organisation. |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

### Service Development

The following requirements pertain to the development of service servers, as opposed to client agent code development, defined next.

##### TR-ID: **Service Language**

|  |  |
| --- | --- |
| **Category** | Technical/Development |
| **Statement** | **Custom server system development MUST** be done using **compiled languages.** |
| **Rationale** | Compiled languages reduce risk by testing software at compilation, and improve performance, reducing infrastructure costs over the service’s lifespan. |
| **Details** | Front end service clients are excluded from this obligation. |
| **Prompts** | … |

##### TR-ID: **Code Development Standards**

|  |  |
| --- | --- |
| **Category** | Technical/Development |
| **Statement** | **Custom system(s)** **MUST** be developed according to **sponsor organisation** provided coding standards. |
| **Rationale** | … |
| **Details** | Reasonable standards are expected to define obligations for: - Documentation - Naming Patterns - Contracts - Integrations - Development Patterns |
| **Prompts** | … |

##### TR-ID: **Code Development**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

##### TR-ID: **Code Testing**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

##### TR-ID: **Time Bound Reference Data**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | Reference Data MUST be describable with Start and End Dates. |
| **Rationale** | In many business domains (e.g.: education), operational change management is simplified and less execution errors occur when change can be done beforehand but scheduled to take effect or terminate at a specific date in the future (e.g.: next term start).  The same for [**role**](#Term_Role) allocations: they may be issued early, but only take effect at a future date (e.g.: beginning of the next month or start of the next term). Both system and [**users**](#Term_SystemUser) provided resources are similar. New material (e.g.: new teaching curriculums and associated resources) may be developed earlier, but only published and made available at a future date. For security reasons, while [**role**](#Term_Role) associations could have undefined end dates for permanent staff, it is not a recommended approach. Instead, always setting an end date and raising reminders to appropriate [**role**](#Term_Role)s that the association is soon coming to an end, permits extending them easily while not leaving risks associated to forgotten ex-employees still having [**role**](#Term_Role)s. |
| **Details** | In single tenancy contexts, the start and end metadata can be in the reference data itself.  In multi-tenancy contexts, where the start and end data may require being within entities that join the tenancy to the reference data type. |
| **Prompts** | … |

**Data Access**

##### TR-ID: **Variables**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

##### TR-ID: **Stored Procedures**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

##### TR-ID: **ORMs**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

### API Development

##### TR-ID: **API**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

### Service Agent Development

The following requirements pertain to the development of client side presentation code.

##### TR-ID: **Service Agent** **Language**

|  |  |
| --- | --- |
| **Category** | Technical/Development |
| **Statement** | **Custom client system development SHOULD** be done using **transpiled languages.** |
| **Rationale** | Transpiled languages (e.g. Typescript) perform checks that are not done until compilation if using JavaScript directly. |
| **Details** | Not all scenarios permit the use of transpiled languages. |
| **Prompts** | … |

### GUI Development

##### TR-ID: **BREAD**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | System Views MUST follow a BREAD **flow.** |
| **Rationale** | Using an intuitive, well-known, repeatable pattern improves learnability and usability of the system while decreasing the use of novel solutions that decrease maintainability and increase risk. |
| **Details** | BREAD is a UI flow pattern, similar to how CRUD is a data access pattern. Browse is a queryable Search/Browse View, that returns a list of search item summaries, any one of which can be clicked to show a user a Read-only view of the singular item, from which they can either return or proceed to an Edit view, from which they can return, save, or simply [logically] Delete the editable record. **Note:** Refer to sponsor organisation available guidance on BREAD. |
| **Prompts** | … |

##### TR-ID: **Input, Output, Containers**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | Interfaces Views **MUST** be developed as Input, Output or nestable Container components. |
| **Rationale** | System logic complexity is reduced when components have a single purpose, and do not try to provide both input and output at the same time. |
| **Details** | Containers can other containers and/or Input or Output Views.  But Input components must not provide Output capabilities and vice versa. |
| **Prompts** | … |

### Messaging Development

Emails sent out MUST be able to configurable in the subject line, and body prefix and footer.

Specifically, this supports the requirement that the message clearly specify its data classification (on the subject line) and optionally instructions in the footer.

## Data Handling

Production data will not be handled in any non-production data environment.

Appendices

Appendix A - Document Information

### Versions

* 1. Initial Draft

### Images

[Figure 1: TODO Image 2](#_Toc144995112)

### Tables

[Table 1: TODO Table 3](#_Toc145048484)

[Table 2: TODO Table 2 3](#_Toc145048485)

### References

**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.

### Structure

Where possible, the document structure is guided by either ISO-\* standards or best practice.

### 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

The following terms and acronyms are used throughout the above Non-Functional Requirements.

#### Conditional Terms

**IF**

: a conditional statement defining the case(s) in which the requirement [**statement**](#Term_Statement) applies – typically defining whether it applies to solutions where the system(s) are [**SaaS**](#Term_SaaS) or [**Custom**](#Term_CustomSystem) or [**OTS**](#Term_OTS) [**SaaP**](#Term_SaaP) solutions.

**THEN**

: the desired outcome of a [conditional](#Term_IF) [**statement**](#Term_Statement).

**ELSE**

: the alternate outcome if the conditional statement is not met.

#### Modal Terms

[Requirement](#Term_Requirement) [statement](#Term_Statement)s are developed as one of the following modals:

**MUST** : an Obligation statement.

**SHOULD** : an Recommendation statement.

**COULD** : an permission statement.

**MUST NOT** : an Prohibition statement.

#### Quantitative Terms

**All**

: a legally ambiguous term to be avoided in …[**all**](#Term_All)…requirements.

#### Domain Terms

Appendix B - FAQs

…

Appendix C – Requirement Record Template

Following guidance within *ITC Project Guidance – Definition – Requirements Development* the schema template for Requirements in this document is as shown below.

##### TR-ID: **Title**

|  |  |
| --- | --- |
| **Category** | Technical/… |
| **Statement** | [**IF**](#Term_IF) … [**THEN**](#Term_THEN) … **ELSE** … |
| **Rationale** | … |
| **Details** | … |
| **Prompts** | … |

1. See ISO-25012 for target user experience qualities. [↑](#footnote-ref-2)
2. See ISO-25010 for target system qualities. [↑](#footnote-ref-3)