**TABLE OF CONTENTS**

1 Introduction 2

1.1 Document overview 2

1.2 Abbreviations and Glossary 2

1.2.1 Abbreviations 2

1.2.2 Glossary 2

1.3 References 2

1.3.1 Project References 2

1.3.2 Standard and regulatory References 2

1.4 Conventions 2

2 Architecture 3

2.1 Architecture overview 3

2.2 Logical architecture overview 3

2.2.1 Software Component 1 description 3

2.2.2 Software Component 2 description 3

2.2.3 Software Component 3 description 3

2.3 Physical architecture overview 3

2.3.1 Hardware Component 1 description 3

2.3.2 Hardware Component 2 description 4

2.3.3 Hardware Component 3 description 4

2.4 Software COTS 4

3 Dynamic behaviour of architecture 5

3.1 Workflow / Sequence 1 5

3.2 Workflow / Sequence 2 5

4 Justification of architecture 6

4.1 System architecture capabilities 6

4.2 Network architecture capabilities 6

4.3 Risk analysis outputs 6

4.4 Human factors engineering outputs 6

5 Requirements traceability 7

# Introduction

## Document overview

This document describes the architecture of ISO 20022 message validation system.

It describes:

* A general description of the system
* The logical architecture of software, the layers and top-level components
* The physical architecture of the hardware on which runs the software
* The justification of technical choices made
* The traceability between the architecture and the system requirements.

## Abbreviations and Glossary

### Abbreviations

Add here abbreviations

COTS: Components Off the Shelf (software industry acronym)

OTSS: Off The Shelf Software (FDA acronym)

SOUP: Software Of Unknown Provenance (IEC 62304 acronym)

## References

### Standard and regulatory References

| # | Document Identifier | Document Title |
| --- | --- | --- |
| 1 | BRS v0 05 | ISO 20022 Message Initiation Validation Tool - BRS v0 05.docx |

|  |  |  |
| --- | --- | --- |
| # | Document Identifier | Document Title |
| [STD1] |  | Add your documents references.  One line per document |

## Conventions

Add here conventions

For example for diagrams.

COTS, OTSS and SOUP refer to the same concept, i.e. software delivered by 3rd party that wasn’t developed with a regulatory and/or normative compliant development process.

We deliberately use the term “SOUP”, to focus on IEC 62304 compliance.

# Architecture

## Architecture overview

Give a general description of the system, from the point of view of the user:

* In what environment it works (home, near patient bed, operating room …)
* Who the users are
* What it is for,
* The main functions,
* The main interfaces, inputs and outputs.

If your software is integrated in a larger system, you may reference a document that describes this system.

## Physical architecture overview

Describe the hardware components on which software runs and their interactions/relationships

Use components diagrams, deployment diagrams, network diagrams, interface diagrams…

### Hardware Component 1 description

Describe the content of each hardware component in the architecture

**Optional**, you may not do it if your software is not class C according to IEC 62304

The description shall contain:

* Its identification
* The purpose of the component
* The software component it receives
* Its technical characteristics: type of machine, CPU, RAM, disk and so on.
* Its network hardware interfaces

### Hardware Component 2 description

Repeat the pattern for each top-level component.

### Hardware Component 3 description

Repeat the pattern for each top-level component.

## Logical architecture overview

Describe the top level software components and their interactions/relationships.

Use UML package diagrams and/or layer diagrams and/or interface diagrams.

Describe also the operating systems on which the software runs.

### Software Component 1 description

Describe the content of each top-level software component in the architecture

**Optional**, you may not do it for 2 rationales:

1. Either your software is not class C according to IEC 62304

2. Or you describe each top level component in a SDD.

The description should contain:

* **Its identification**
* The purpose of the component,
* Its interfaces with other components,
* Its network interfaces,
* The hardware resources it uses, for example: average RAM usage, peak RAM usage and peak frequency and duration, disk space for permanent data, disk space for cache data, average CPU usage, peak CPU usage and peak frequency and duration …

### Software Component 2 description

Repeat the pattern for each top-level component.

### Software Component 3 description

Repeat the pattern for each top-level component.

## Software SOUP

If you use SOUP (Software Of Unknown Provenance), list them here.

For each SOUP, describe:

* Its identification and version
* Its purpose
* Where it comes from: manufacturer, vendor, university …
* Wether it is maintained by a third party or not
* If this is an executable,
  + What are the hardware / sotfware resources it uses
  + Wether it is insulated in the architecture and why
* Its interfaces and data flows
* Which SOUP functions the software uses
* How the SOUP is integrated in the software
* What hardware/software resources it requires for proper use

Note: have a look at FDA Guidance « Off-The-Shelf Software Use in Medical Devices » to determine if you need specific or special documentation for your COTS.

If there is a list of known bugs on your COTS, you may add here this list with a review of their consequences in terms of software failure and patient safety. If there are concerns about known bugs, they should be treated by the risk analysis process.

# Dynamic behavior of architecture

The architecture was designed to answer to functional requirements.

For each main function of the system, add a description of the sequences / data flow that occur.

Use sequence diagrams, collaboration diagrams

## Workflow / Sequence 1

Describe here the workflow / sequence of a main function

For example, the user queries data, what happens, from his terminal to the database.

## Workflow / Sequence 2

Repeat the patern for each main function of the system

# Justification of architecture

## System architecture capabilities

Describe here the rationale of the hardware / software architecture in terms of capabilities:

* Performances (for example response time, user mobility, data storage, or any functional performance which has an impact on architecture)
* User / patient safety (see §4.3 and §4.4)
* Protection against misuse (see 4.4)
* Maintenance (cold maintenance or hot maintenance),
* Adaptability, flexibility
* Scalability, availability
* Backup and restore
* Hardware and Software security : fault tolerance, redondancy, emergency stop, recovery after crash …
* Administration,
* Monitoring, audit
* Internationalization

## Network architecture capabilities

If the medical device uses/has a network, describe here the rationale of the hardware / network architecture:

* Bandwidth
* Network failures
* Loss of data
* Inconsistent data
* Inconsistent timing of data
* Cyber security (see FDA Guidance on Cyber Security of networked medical devices)

## Risk analysis outputs

If the results of risk analysis have an impact on the architecture, describe here for each risk analysis output what has been done to mitigate the risk in the architecture.

Use diagrams if necessary, like architecture before risk mitigation and architecture after risk mitigation, to explain the choices.

## Human factors engineering outputs

If the results of human factors analysis have an impact on the architecture, describe here for each risk human factors output what has been done to mitigate the risk in the architecture.

## SOUP integration

If the software architecture has a particular structure dedicated to SOUP integration, it can be described here. For example a wrapper of the SOUP, or an external process + a socket communication, …

# Requirements traceability

Add a table with traceability of components of this document with functional requirements.

|  |  |  |
| --- | --- | --- |
| Requirement | Component | Comment |
| REQ-001  The device shall do foo | COMPO-001: foo maker | COMP-001 does foo.  COMP-002 also does verification of foo result. |

This may be a difficult job. A high level function is usually handled by many components. In this case, quote only the component(s) which has(have) the major role.