

---

**Product name**  
**Functional Specification Template**

**Revision: 0.2**

Authors: Syd Chapman  
Juri Papay

---

Open Middleware Infrastructure Institute  
Room 6005, Building 21 (Faraday)  
Highfield Campus  
University of Southampton  
Hampshire, SO17 1BJ  
Tel: +44 (0)2380 58787

---

Revision No.	Author	Changes Made	Date Revised
0.1	Syd Chapman	IBM functional specification template	date
0.2	Juri Papay	Update FS	10.03.2005
0.3	Juri Papay	Update FS	25.05.2005
0.4			
0.5			
0.6			
0.7			
0.8			

**Table of Contents**

1 *Introduction* ..... 4

2 *Product description* ..... 4

3 *Use cases* ..... 5

4 *Evaluation metrics* ..... 5

5 *Functional design considerations* ..... 5

6 *List of functions* ..... 6

# 1 Introduction

The purpose of this document is to present a template of a Functional Specification as guidance for the partners of OMII Managed Programme. The main purpose of a Functional Specification (FS) is to describe the functionality and the use of the product (the software) from the user's perspective, i.e. how the user is going to use the product and what are the interactions between the product and the user. The FS doesn't detail the architecture of the product or how it is going to be implemented. The FS reflects a collective understanding of the product. The most important requirement to a FS, apart from being correct is to be understandable and easy to read. Therefore it is important to write good specifications that are read and reviewed by many people. It is also important to frequently revise the FS and to make sure that the content reflects the changes and new issues that came up during product development.

The benefits of a functional specification can be summarised by the following points:

- easier product development
- the FS will be approved by a client (in this case the OMII) , to give confidence that the product development is on the right track
- easy to interpret, minimise communication problems and misunderstanding
- the document is written in a human language, it is easier to communicate design ideas and change a few paragraphs rather than *deciphering* a massive amount of code.
- easier to find alternative solutions
- if the FS is available it is easier to allocate resources
- introduces an element of discipline in the software development process
- FS can be used for validation purposes, i.e. for checking whether the product really implements the features described in the FS
- form the basis for testing and verification

In the following section we provide a skeleton that can be used for writing a FS document.

## 2 Product description

Describes the motivation for product development, and lists the most important features and capabilities.

This section should also provide a brief analysis of similar products. Here are some questions that can aid the analysis:

What problems do the similar products have?

Which of these problems does the product try to solve?

How will this product be an improvement?

What is the added value?

### **3 Use cases**

Use Cases describe the sequence of events that represent the interaction between the user and the product. These events can be described as pairs of actions performed by the user and the responses of the product. This section should also detail security scenarios and robustness i.e. how the system will react to faulty/invalid inputs or incorrect usage. The specification of use cases is an important requirement since it helps to identify the detailed functionality of the product.

User characteristics - describe who will be using the product, i.e. the main characteristics of the user in terms of role and specific knowledge. For example there might be different use scenarios for research staff, system administrators, project managers and these scenarios may also vary according to the affiliation to different groups.

### **4 Evaluation metrics**

The Evaluation metrics enables to measure the efficiency of the product? For example response time, task throughput, number of messages processed in a notification system, time saving for the user, better colours etc.

### **5 Functional design considerations**

Functional design considerations describe the operating environment for the product and details the product's attributes that affected the product's functional design. This section should detail the following aspects:

- Assumptions that were made during functional design
- Prerequisites for the correct working of the product
- Main decisions/reasons regarding functionality
- Scale of deployment
- Resource requirements in terms of hardware, data volume, other software or equipment
- Portability
- Reliability/Maintainability/Availability
- Installation
- Security
- Configuration and customisation
- Error handling

## 6 List of functions

The list of functions is the largest and the most important section of the FS document.

This section defines the complete list of product's functions with the associated input/output arguments.

This can be a fully text-based section or you can include diagrams or tables for each individual function as shown below – whatever is most efficient to explain the functionality.

The functions should be grouped into interfaces and associated with modules. In this respect the FS should detail the following issues:

- Identify modules into which the product will be partitioned
- Associate interfaces with modules
- Group the functions into interfaces
- Describe the data/control flow
- Identify the data structures

Each function should be described in following terms:

- **Description** – the purpose of the function.
- **Input arguments** – input format, module that supplies the input, range of valid inputs
- **Process** – describes the main steps in pseudocode performed by the function.
- **Output** – desired output and format, destination for the output i.e. identify the module where the output propagates.
- **Exceptions** – describe situations when exceptions can occur and the exception handling procedures.
- **Comments**

For example the function can be described in a tabular format:

<b>Description</b>	Building Axis from source files
<b>Input arguments</b>	ant-script
<b>Process</b>	Ant tool executes the instructions contained in the ant-script
<b>Output</b>	Display Messages: “Start building Axis ... ... Build Axis complete.”  As a result of this operation the source files are compiled and put in the /axis-1_2beta/build directory.
<b>Exceptions</b>	A failure is indicated by “Build failed” message. This may be due to incorrect version of Ant, insufficient disk space, no write access, error in ant-script.
<b>Comments</b>	-