NWU® | Information Technology

FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

1 Requirements of an IT System

The end goal of a project is to deliver a high-quality product precisely as the customer requested. Functional requirements are the primary way that a customer communicates their needs to the project team. Functional requirements help to keep project team moving in the right direction.

Understanding the difference between functional and non-functional requirements will help both the client and the IT supplier as they will understand their requirements clearly. This clarity leads to scope refinement, optimised cost, and finally a happy customer.

If there is one thing any project must have to prevent failure, it is a sensible and comprehensive collection of both the functional and non-functional requirements.

1.1 The difference between Functional and Non-functional requirements

The official definition of a 'functional requirement' is that it essentially **specifies something the system should do.**

Simply put, the difference is that **non-functional requirements describe how the system works**, while **functional requirements describe what the system should do**.

The definition for a non-functional requirement is that it essentially specifies **how the system should behave** and that it is a constraint upon the system's behaviour. One could also think of non-functional requirements as to **quality attributes** for a system.

If the functionality of the product is not dependent on non-functional requirements, then why are they important? **The answer is in-usability. Non-functional requirements affect the user experience as they define a system's behaviours, features, and general characteristics**.

Non-functional requirements, when defined and executed well, will help make the system easy to use and enhance performance.

1.2 Basically, functional requirements can be divided into 4 groups which are

1.2.1 Business requirements**: They contain the goal, such as an order system, an online catalogue, or a physical product. It can also include things like approval workflows and authorisation levels.**

1.2.2 Administrative functions: **They are the routine things the system will do, such as reporting.**

1.2.3 User requirements: **They are what the system's user can do, such as place an order or browse the online catalogue.**

1.2.4 System requirements: **These are things like software and hardware specifications, system responses, or system actions.**

1.3 Once the functional requirements are defined, then it's time to think about the non-functional requirements, such as

- **Usability**: This focuses on the appearance of the user interface and how people interact with it. What colour are the screens? How big are the buttons?
- **Reliability / Availability:** What are the uptime requirements? Does it need to function 24/7/365?
- **Scalability**: As needs grow, can the system handle it? For physical installations, this includes replacement hardware or space to install it in the future.
- **Performance:** How fast does it need to operate?
- **Supportability:** Is support provided in-house, or is remote accessibility for external resources required? Long term viability of outsourcers.
- The **reliance on key personnel** over the lifetime of the solution.

- **Security:** What are the security requirements, both for the physical installation and from a cyber-perspective?
- **Capacity:** Disk space, human resources, memory, etc. – requirements.
- **Recoverability:** How quick can you recover from a disaster, and what are the requirements.
- **Maintainability:** Updates**,** Upgrades and maintenance, including version control.
- **Serviceability**: Long term service defined by SLA
- **Regulatory**: Does it meet all regulatory requirements, like POPIA
- **Manageability**: How do you manage the software, including contractual agreements, SLA's, licenses, etc.
- **Data Integrity:** How well the software system maintains the data in terms of accuracy, authenticity, and without corruption.
- **Usability**: How easily the user can learn, operate, prepare inputs and interpret outputs through interaction with a software system. Are there sufficient documentation available?
- **Interoperability and Integration**: Refers to the basic ability of computerised systems to connect and communicate with one another readily.
- **Architecture: D**oes it fit into the NWU-IT Architecture. Cloud, on-prem, hybrid, SAAS, ….
- **Licensing:** Are the terms explicit and is it understood? Who will manage the licenses?
- **SLA's:** For all system development projects, there should be an SLA between client and supplier.
- **Contracts:** Are there contracts involved? Legal office involved?
- **The total cost of ownership**: Cost is not just about the initial purchase and implementation. There is a lifecycle cost that could be ten times the initial cost.
- **Backup and recovery procedures**