UNIVERSIDAD CENTRAL DEL ECUADOR

Omnium potentior est sapientia

Functional and non-functional requirements of a Program

Information Systems Engineering

Dilan Ismael Caizapasto Villarreal,

Daniel Andres Caizatoa Torres

Parallel “01”

Ing. Juan Pablo Guevara Gordillo

24 jun. 24

Description

Requirements set the expectations and functionalities of a program in the field of software development. Most often these requirements are split into two main groups: functional and non-functional requirements. It is crucial to identify and document these requirements as failure to capture them leads to a product that does not make much sense for end users, and thus stakeholders.

Functional and non-functional requirements are crucial elements of the software development process due to their necessary overlapping, which work together as a great framework for capable of building efficient systems that can satisfy business needs in an efficient way.

The main objective of this task is to make known the concept of what functional and non-functional requirements are in the field of programming. In addition, five functional and five non-functional requirements that are part of the wallet program in Java will be analyzed.

Functional requirements

Software engineering functional requirements specify what the system should do, detailing the tasks it should perform, and are built to meet user needs. This behavior includes things like inputs, processes performed on the data, and system outputs, A system's external interactions (how a user interacts with the system or how the system talks to other systems). It is a guide for the design, implementation, and testing of software applications.

According to Ruth Malan and Dana Bredemeyer:

> Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. In product development, it is useful to distinguish between the baseline functionality necessary for any system to compete in that product domain, and features that differentiate the system from competitors' products, and from variants in your company's own product line/family. Features may be additional functionality, or differ from the basic functionality along some quality attribute (such as performance or memory utilization). One strategy for quickly penetrating a market, is to produce the core, or stripped down, basic product, and adding features to variants of the product to be released shortly thereafter. This release strategy is obviously also beneficial in information systems development, staging core functionality for early releases and adding features over the course of several subsequent releases. (Ruth & Dana, 2001)

Non-functional requirements

In software engineering, non-functional requirements (NFRs) define how well a system performs its functions, they are also known as "quality attributes" - in contrast to Functional Requirements (see the sidebar). Their perspective is also typically found in functional requirements, which drive the operational behavior of the system and thereby have established levels of performance and user acceptability. They help design and discuss the system to ensure the quality and efficiency of the product.

According to Mohammad Dabbagh and Sai Peck Lee:

> Functional requirements describe the functional behavior of the system whereas nonfunctional requirements express how good a system should work. It has been widely acknowledged that a quality attribute such as reliability, modifiability, performance, or usability is a nonfunctional requirement of a software system. Although functional and nonfunctional requirements are very different, they have a serious impact on each other. Several studies, for example, stated that the achievement of nonfunctional require-ments along with functional requirements is critical to the success of a software system. (Mohammad Dabbagh & Sai Peck Lee, 2014)

Five Functional Requirements of Wallet's Program

1. *Transaction Registration.* The system must allow new purchase and sale transactions to be registered by entering an amount in the text field and pressing the corresponding buttons.
2. *Total Calculation.* The system must calculate and display the accumulated total of transactions in real time, adding purchases and subtracting sales.
3. *Transaction Display.* The system must display a table with all recorded transactions, including ID, type (purchase or sale), date and amount.
4. *Date and Time Update.* The system should automatically record and display the date and time each transaction was made.
5. *Transaction Search.* The system must allow users to search for specific transactions by entering a search criteria in the text field.

Five Non-functional Requirements of Wallet's Program

1. *Usability.* The system should have an intuitive and easy-to-use graphical user interface (GUI), with clearly labeled and accessible buttons.
2. *Performance.* The system should update and display the running total and transaction table in less than 1 second after recording a new transaction.
3. *Security.* The system must ensure that only authenticated users can record, modify or delete transactions.

4. ***Maintainability.*** The system code must be well documented and structured to facilitate future updates and maintenance.
5. ***Compatibility.*** The system must be compatible with multiple operating systems, including Windows, macOS, and Linux.

Conclusions

- ***Requirements matter.*** Functional along with non-function requirements are important in software development. They specify the behaviors and characteristics that a system must exhibit to realize its functionality, supply users' demands, and satisfy stakeholder expectations.
- ***CF is Complementary Function.*** Applications requirements and non-functional requirements are complementary to each other. Functional requirements say what the system will do, non-functional requirements explain how well these functions are performed. Together, these ensure a rich foundation for building efficient and user-working systems.
- ***Impact on system success.*** The potential success or failure of a software systems hangs precipitously upon the correct identification and development of these two categories. Focusing solely on one will result in the system missing user needs or underperforming, while the other is discarded.

Recommendations

- ***Comprehensive requirements gathering.*** *Spend enough time in the initial stages of development to comprehensively collect and document functional and non-*

functional requirements. Involve stakeholders and end users in this process to ensure all needs and expectations are captured.

- ***Requirements Prioritization.*** *Prioritize requirements based on their importance and impact on the success of the system.*
- ***Documentation and Maintainability.*** Maintain thorough documentation and adhere to coding standards to facilitate future updates and maintenance. Well-documented code ensures that the system can evolve with changing requirements and technologies.

Bibliography

Malan, R., & Bredemeyer, D. (2001). Functional requirements and use cases. *Bredemeyer Consulting*, 335-1653.

Dabbagh, M., & Lee, S. P. (2014). An approach for integrating the prioritization of functional and nonfunctional requirements. *The Scientific World Journal*, *2014*(1), 737626.