Bellevue University

Distinction between User Requirements and System Requirements

& Change

Timothy Jelinek

CS430-T301 Software Engineering

3/17/2024

**Part 1**

According to JD(2008), “User requirements are statements about what the user needs the system to do to solve a specific problem or meet a specific need.” User requirements are written in the language the user understands, while the system requirements are technical descriptions saying how the system is to be built to meet the user requirements. It is important to make a distinction between the user and system requirements so that you can get better outcomes in software development projects through creating precision around the perspective, making effective trade-offs, and ensuring the appropriate lens is used when needed. User requirements are important to state the problem or need the user wants the system to accomplish. System requirements are important to define the technical aspects like system components, interfaces, performance, security, and other technical aspects that are needed to build the system. Failure to capture and distinguish the user requirements from the system requirements can cause miscommunication, wasted effort, and project failure. These possible consequences show that it is vital to know the differences between the user requirements and system requirements for the software project to be successful. I’ve learned through reading this article how important user requirements and system requirements are and that they should be distinguished so that it is easier to create software projects and have a clear focus and plan in mind when starting the projects. Before reading this article, I didn’t fully comprehend or know the action in separating the user requirements from the system requirements and how big of a role this played in software projects.

**Part 2**

Software change is inevitable for many reasons, such as new requirements emerge when the software is used, the business environment changes, errors need to be repaired, new computers and equipment are added to the system, and the performance of the system needs to be improved. Sommerville, Software Engineering(2014) states, “Organizations have huge investments in their software systems - they are critical business assets. To maintain the value of these assets to the business, they must be changed and updated. The majority of the software budget in large companies is devoted to changing and evolving existing software rather than developing new software.” There are software process activities you can use to predict changes and make the software being developed more resilient to change, which include embracing modularity, applying design patterns, following SOLID principles, implementing testing and automation, and adopting agile and iterative methods. According to the LinkedIn community(2024) “One of the key aspects of designing resilient systems is to break down complex problems into smaller, manageable, and independent components.” By breaking complex problems into smaller ones, you are able to isolate the impact of change and enable parallel development and testing. Applying design patterns allow you to keep a more readable system, and by having these patterns in place, you don’t have to continuously spend time reinventing the wheel. The SOLID principles are single responsibility, open-closed, Liskov substitution, interface segregation, and dependency inversion. Using these principles helps in improving the cohesion, coupling, abstracting, and modularity of the code you are writing.

Sources:

Jd. (2023, March 16). *Why Differentiate User Requirements vs. System Requirements? - Shaping Software*. Shaping Software. https://shapingsoftware.com/user-requirements-vs-system-requirements/

*How can you design software systems that are resilient to change?* (2024, January 28). www.linkedin.com. https://www.linkedin.com/advice/1/how-can-you-design-software-systems-resilient-change-goyrf

*CS 410/510 - Software Engineering class notes*. (n.d.). https://cs.ccsu.edu/~stan/classes/CS410/Notes16/09-SoftwareEvolution.html