**Question-1)** **For each type of Non Functional Requirements give two example each and identify Goal, Objective and Quantitative verifiable metric where possible to create proper non-functional statements.**



**Usability:**

**Goal:** An understanding of the five characteristics of usability – **effective, efficient, engaging, error tolerant, easy to learn** – helps guide the user-centered design tasks to the goal of usable products

**Objective:** The most effective usability objectives contain several key elements:

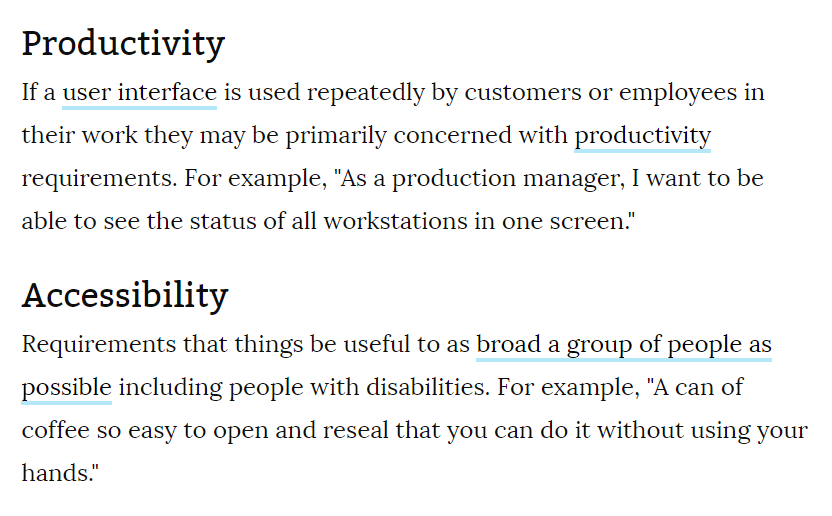
* Task – including a quantitative goal so that success can be easily defined, usually in terms of time, accuracy, task completion rate, etc. The highest risk tasks and the most important tasks for using the device effectively should be included when defining tasks.
* User profile – specific user group that will perform the task.
* Use environment – context of device use for a particular task, including physical and social environmental factors. Use environment is not always specifically called out in the usability objective but is often implied. It should, at a minimum, be defined for the device in the usability documentation and considered when developing usability objectives.

**Quantitative verifiable metric:**

**The most basic measures are based on the definition of usability as a quality metric:**

* success rate (whether users can perform the task at all),
* the time a task requires,
* the error rate, and.
* users' subjective satisfaction

**Examples:**



**Reliability:**

**Goal: To use engineering knowledge and techniques to prevent certain failure modes and to reduce the likelihood and frequency of failures**. To identify and correct the causes of failures that do occur, despite the efforts to prevent them.

**Objective:**  The objectives of reliability engineering, in decreasing order of priority, are: **To apply engineering knowledge and specialist techniques to prevent or to reduce the likelihood or frequency of failures**. To identify and correct the causes of failures that do occur despite the efforts to prevent them

**Quantitative verifiable metric:**

A commonly used metric used to measure asset reliability is using its Failure Metrics, which include **Mean Time Between Failure (MTBF), Mean Time to Failure (MTTF), and Mean Time to Repair (MTTR)**.

**Example:**

**If you are using an ATM machine, insert the card into the machine, type your password, ask the machine to withdraw an amount of 100$. The amount is delivered by machine but you forgot to pick up the cash from the machine and move it outside of the ATM room. Now, its reliability requirement of the ATM software to rollback the money.**

Another simple example of reliability is **an alarm clock that rings at 7:00 each morning, but is set for 6:30**. It is very reliable (it consistently rings the same time each day), but is not valid (it is not ringing at the desired time).



**Portability:**

**Goal:** Software that requires much environment-related configuration and tuning will cost time and effort as new versions move through the life cycle. Portability **saves time and mental overhead for anyone involved in moving new versions of the software across environments**.

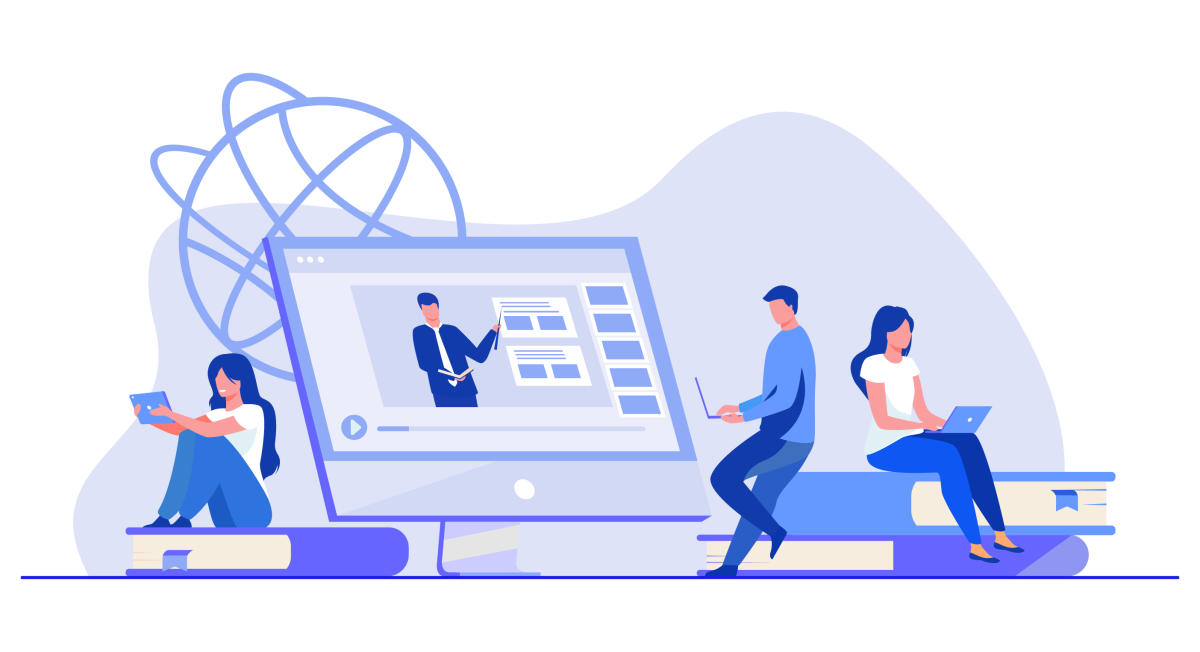
**Objective:** The objective of the portability testing is: **To determine whether a system can be ported to each of the environmental characteristics**.

**Quantitative verifiable metric:** Portability, in relation to software, is **a measure of how easily an application can be transferred from one computer environment to another**. A computer software application is considered portable to a new environment if the effort required to adapt it to the new environment is within reasonable limits.

**Example:**

**A program running on Windows 10 must be able to run on Windows 11 without any change in its behavior and performance**

A term used to describe an object that can be easily moved, such as a portable computer. For example, **a laptop** is a good example of a portable computer. 2. When referring to computer hardware, portability describes an external device that can be moved from one place to another without disrupting its operation.



**Implementation:**

**Goal:** The primary activities of software implementation include the: **Fabrication of software units to satisfy structural unit specifications**. Assembly, integration, and testing of software components into a software configuration item.

**Objective:** Software implementation refers to the process of **adopting and integrating a software application into a business workflow** (as a part of a digital transformation strategy). Implementation of new tools and software into an enterprise can be complex, depending on the size of the organization and the software

**Quantitative verifiable metric:** Process metrics are **the measures of the development process that creates a body of software**. A common example of a process metric is the length of time that the process of software creation tasks.

**Example:**

strategic implementation within a business context might involve **developing and then executing a new marketing plan to help increase sales of the company's products to consumers**.

**Standard Requirement:**

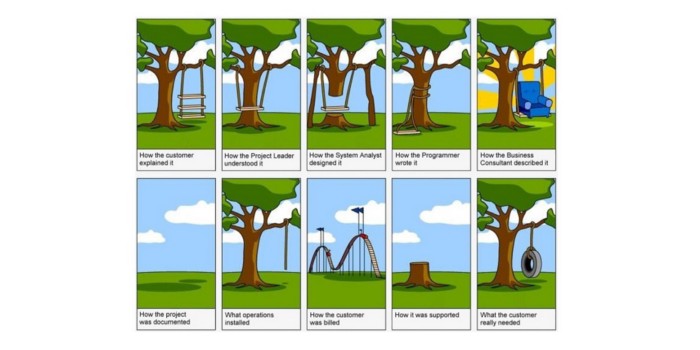
**Goal:**  The goal of requirement engineering is **to develop and maintain sophisticated and descriptive 'System Requirements Specification' document**

**Objective:** Objectives, or design goals, are **the desired attributes of the design, what the design will "be" and what qualities it will have**. They are often adjectives and characterized by present participles such as "are" and "be." Ideally, they are separated into primary and secondary objectives

**Quantitative verifiable metric:** Process metrics are **the measures of the development process that creates a body of software**. A common example of a process metric is the length of time that the process of software creation tasks

**Example:**

 you iterate first on the user requirements; elicitation, specification, and validation, and repeat the same steps for the system requirements.



functions offered by the system such as timing constraints, constraints on the development process, standards, etc. e.g. **reliability, response time and storage requirements**. Constraints are I/ O device capability, system representations, etc

**THE END**