**TESTING OF SOFTWARE  
CHARACTERISTICS**

**UNIT V**

**INTRODUCTION:**

* A standard definition of the quality of a product is the degree to which the product fulfills the expectations the customers and users have for it.
* Quality attributes represent a way of structuring and expressing the expectations for a product.
* Testers to understand what quality attributes are and how they may be expressed.
* With this understanding testers can contribute to the quality of a product in a number of ways from the very start of the development life cycle.
* The quality of a product should be measured both
  + What the product shall do
  + How the functionality shall present itself and behave
* The *ISO 9126 standard, a standard providing a quality model for product* quality attributes.
  + Functionality
  + Reliability
  + Usability
  + Efficiency
  + Maintainability
  + Portability
* The test analyst is concerned with the functional and the usability quality attributes.
* The technical test analyst is concerned with the other four quality attributes
* ISO 9126 expresses that compliance to relevant standards and regulations

**QUALITY ATTRIBUTES FOR TEST ANALYSTS:**

* The functionality is what the product can do. Without functionality we don’t have a product at all.
* Functionality may be tested at almost all test levels
* ISO 9126 breaks the functionality attribute into:
  + Suitability
  + Accuracy
  + Interoperability
  + Security

**Functional Testing:**

**a)Suitability Testing:**

* In suitability testing we test the requirements or needs concerned with the presence and appropriateness of a set of functions for specified tasks and user objectives
* Suitability of a product is about how the functionality of the software product supports the tasks the users are performing
* what detailed suitability could concern is:
  + Data availability (i.e., how do we get data, both in terms of background or reference data and/or data from other system information and in terms of data input and change facilities and the associated levels of data validations)
  + Data handling (i.e., what is data used for, for example, as event-driven interrupts or signals and—not least: calculations, actions, and so forth based on data and other input)
  + Result presentation (i.e., output facilities, for example, in terms of windows and reports)
* Suitability testing can take place at all testing levels.

**b)Accuracy Testing :**

* In accuracy testing we test the requirements or needs concerned with the product’s ability to provide the right or agreed upon results or effects.
* A more detailed accuracy specification could concern:
  + Algorithmic accuracy: Calculation of a value from other values and the correctness of function representation
  + Calculation precision: Precision of calculated values
  + Time accuracy: Accuracy of time related functionality
  + Time precision: Precision of time related functionality
* If in doubt about these attributes during testing it is better to ask, rather than to assume, especially if you are not an absolute domain expert.
* Accuracy can be tested at all testing levels, the earlier the better.

**c)Interoperability Testing**

* In interoperability testing we test the requirements or needs concerned with the ability of our software system to interact with other specified systems.
* No software system stands alone; it will always have to interact with other systems in the intended deployment environment, such as hardware, other software systems like operating systems, database systems, browsers, and be-spoken systems, external data repositories, and network facilities.
* Detailed interoperability could concern:
  + Inbound interoperability: Ability to use output from standard, third party, or in-house products as input
  + Outbound interoperability: Ability to produce output in the format used by standard, third-party, or in-house products
  + Spawnability: Ability to activate other products
  + Activatability: Ability to be activated by other products
* Interoperability testing usually takes place at the system integration testing level.

**d)Functional Security Testing**

* In security testing we test the requirements or needs concerned with the ability to prevent unintended access and resist deliberate attacks intended to gain unauthorized access to confidential information, or to make unauthorized modifications
* Detailed security could be concerning:
  + Activity auditability: Log facilities for activities, actors, and so forth
  + Accessability: Access control mechanisms
  + Self-protectiveness: Ability to resist deliberate attempts to violate access control mechanisms
  + Confinement: Ability to avoid accidental unauthorized access to facilities outside the application
  + Protectiveness: Ability to resist deliberate attempts to access unauthorized facilities outside the application
  + Data integrity: Protection against deliberate damage of data Data privacy: Protection of data against unauthorized access
* Security testing can be split into functional security testing and technical security testing.

**Usability Testing:**

* Usability is the suitability of the software for its users, in terms of the effectiveness, efficiency, and satisfaction with which specified users can achieve specified goals in particular environments or contexts of use.
* The *effectiveness of a software product is its capability to enable users to* achieve specified goals with accuracy and completeness.
* The *efficiency of a* product is its capability to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved.
* *satisfaction of a product* is its capability to satisfy users.

**Users Concerned with Usability**

* The usability attribute is related to users.
* A user group for a product is a group of people who will be affected in similar ways by the product.
* This group may indeed be divided into frequent users, occasional users, and rare users, or other relevant subgroups
* For each of the user groups it is necessary to look at different characteristics.
* Age (e.g., preschool, children, teens, young adults, mature adults,
* and elderly)
* Attitude (e.g., hostile, neutral, enthusiastic)
* Cultural background
* Education (e.g., no education yet, illiterate, basic education, middle
* education, workman, university education)
* Disabilities (e.g., people who are dyslexic, color-blind, blind, partially sighted, deaf, mobility-impaired, or cognitively disabled)
* Gender
* Intelligence

**Usability Subattributes**

* ISO 9126 breaks the usability attribute into the following subattributes:
  + Understandability
  + Learnability
  + Operability
  + Attractiveness
* *Understandability has got to do with how difficult it is to recognize the* logical concept and find out how to apply it in practice
  + Extent to which the system maps the concepts employed in the business procedures
  + Extent to which existing nomenclature is used
  + Nature and presentation of structure of entities to work with
  + Presentation of connections between entities
* *Learnability concerns the learning curve for the product*
  + Extent to which a user of the system can learn how to use the system without external instruction
  + Presence and nature of on-line help facilities for specified parts of the system
  + Presence and nature of off-line help facilities for specified parts of the system
  + Presence and nature of specific manuals
* *Operability is about what the product is like to use and control in deployment.*
  + Presence and nature of facilities for interactions with the product
  + Consistency of the man-machine interface
  + Presence, nature, and ordering of elements on each form
  + Presence and nature of input and output formats
  + Presence and nature of means of corrections of input
  + Presence and nature of navigational means
  + Number of operations and/or forms needed to perform a specified task
  + Format, contents, and presentation of warnings and error messages
  + Presence and nature of informative messages
  + Pattern of human operational errors over stated periods of time under
  + stated operational profiles according to defined reliability models
* *Attractiveness has got to do with how the users like the system and what* may make them choose to acquire it in the first place. This may cover the:
  + Use of colors
  + Use of fonts
  + Use of design elements, such as drawings and pictures
  + Use of music and sounds
  + Use of voices (male and female), languages, and accents
  + Layout of user interfaces and reports
  + Presence and nature of nontechnical documentation material
  + Presence and nature of technical documentation material
  + Presence and nature of specified demonstration facilities
  + Presence and nature of marketing material

**Accessibility**

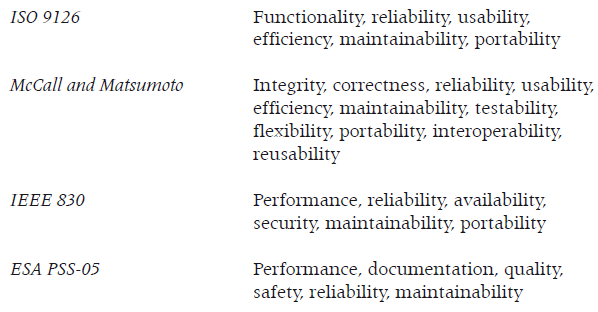
* In recent years there has been more and more focus on equal opportunities, not the least of which are for people with disabilities.
* This also concerns software systems, which must be accessible and operable for everybody
* Accessibility may cover the:
  + Use of colors, especially mixtures of red and green
  + Possibility of connecting special facilities, such as speaker reading the text aloud, Braille keyboard, voice recognition, and touch screens
  + Possibility of using the product entirely by key strokes and/or voice commands
  + Facilities for multiple key pressure using only one finger or other pointing device
  + Possibility of enlarging forms and/or fonts
  + Navigation consistency
* **Establishing Usability Requirements**
* Usability requirements should be expressed as explicitly as possible.
* Usability requirements can be derived from usabilityassessments
* Usability assessment is a requirements elicitation technique—and it should be performed early.
* A usability assessment is performed by representative users who are given tasks to complete on a prototype of the products.
* This can be hand-drawn
* Sketches of forms or mock-ups of the forms made in, for example, powerpoint.
* Any thoughts and difficulties the users have in completing the tasks are recorded.
* After the usability assessment the comments are analyzed; the prototype may be changed and assessed again; and finally the usability requirements are derived.
* Usability assessments can be done very primitively by review of prototypes and storyboards

**Testing Usability**

* Usability may be tested in various ways during the development life cycle.
* Techniques to use may be:
  + Static tests
  + Verification and validation of the implementation
  + Surveys and questionnaires
* *Static tests can be performed as reviews and inspections of usability specifications.*
* The *verification and validation of the implementation of the usability* requirements are performed on the working system.
* *Surveys and questionnaires may be used where subjective measures, such as the percentage* of representative future users who like or dislike the user interface, are needed.

**QUALITY ATTRIBUTES FOR TECHNICAL TEST ANALYSTS:**

* The functionality will always behave and present itself in certain ways.
* we call the non-functional or functionality- sustaining attributes
* Some classics standards, which have been around for quite some time, are listed here with the quality attributes they include:



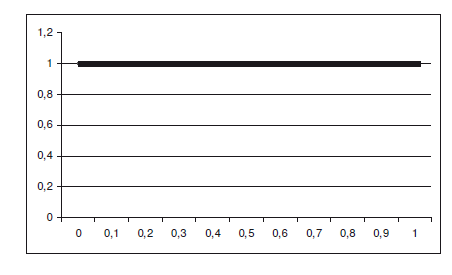
* The nonfunctional requirement types covered in this section are:
  + Reliability
  + Efficiency
  + Maintainability
  + Portability

**Technical Testing in General:**

* In principle the non-functional testing is identical to the functional testing; it should be based on requirements and needs and use the test case design techniques.
* Testers should, help developers and analysts define these requirements from the beginning; and we can review the requirements to ensure that they are comprehensive and testable.
* It is important that the non-functional requirements are measurable and testable.
* The circumstances under which the goal is to be achieved must also be specified.
* It must be remembered that each non-functional requirement must be expressed using a scale, a specific goal, and possibly also acceptable limits.
* The testing of the nonfunctional quality attributes must be executed in a realistic environment reflecting the specified circumstances.
* This can be in terms operational profiles.
* An operational profile is a description of:
  + How many
  + Of which user groups
  + Will use what parts of the system
  + When
  + How much and/or how often
* If care is not taken to ensure realistic circumstances the testing may be a complete waste of time and, even worse, create a false sense of confidence in the product.

**Random Input Technique:**

* The random input technique can assist in generating input data based on a model of the input domain that defines all possible input values and their operational distribution.
* Random input follows the input distribution; the input values are constrained and guided by this.
* Expected input patterns can be estimated or may be known before deployment, and that knowledge can be used in testing.
* In uniform distribution the probability of each value in the value domain is equal.



* For random testing we select input values for the test cases randomly from the input domain according to the input distribution.
* If the distribution is unknown, we can always use a uniform distribution.
* Random input is mainly interesting when the objective is to get the system to crash.
* A large number of test cases can be generated quite quickly, and long sequences of input can be run.
* The expected result and the actual result are, however, usually not that important when we are performing reliability or performance testing.
* The benefit of random input is that it is very cost-effective, especially if automated
* Another benefit is that random input testing may find “unexpected” combinations and sequences and may detect initialization problems.

**Technical Security Testing:**

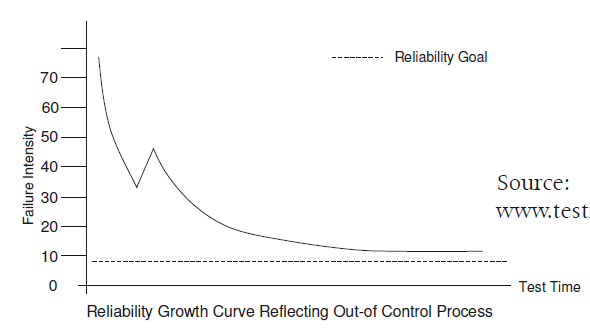
* In security testing we test the requirements or needs concerned with the ability to prevent unintended access and resist deliberate attacks intended to gain unauthorized access to confidential information.
* Detailed security attributes may be:
  + Activity auditability: Log facilities for activities, actors, and so on
  + Accessability: Access control mechanisms
  + Self-protectiveness: Ability to resist deliberate attempts to violate
  + access control mechanisms
  + Confinement: Ability to avoid accidental unauthorized access to
  + facilities outside the application
  + Protectiveness: Ability to resist deliberate attempts to access
  + unauthorized facilities outside the application
  + Data integrity: Protection against deliberate damage of data
  + Data privacy: Protection of data against unauthorized access
* Security testing can be split into functional security testing and technical security testing

**Reliability Testing:**

* Reliability is the probability that software will not cause the failure of a system for a specified time under specified conditions.
* Concerning reliability we have got to be realistic: It is impossible to produce 100% fault-free products!
* Functional testing and reliability testing are connected. The goal of functional testing is to obtain the highest possible reliability of the product within the given limits.
* The reliability testing should be based on operational profiles specified for the product, and reliability goals expressed in requirements.
* A specific reliability may be used as a test completion or test exit criterion for the system testing, allowing for earlier reliability testing
* The ISO 9126 standard breaks the reliability attributes down into a number of sub attributes. There are:
  + Maturity
  + Fault tolerance (robustness)
  + Recoverability

**Maturity Testing:**

* Maturity is the frequency of failures as a result of faults in the software.
* A product’s expected maturity is often expressed in terms of
  + Mean time between failures (MTBF)
  + Failures per test hour
  + Failures per production time (typically months)
  + Failures per number of transactions
* The results of the reliability testing can be graphic presentations of the measurements using a reliability growth model



**Robustness Testing:**

* Fault tolerance or robustness is the product’s ability to maintain a specified level of performance in the presence of software defects or infringement of a specified interface.
* This may cover the product’s:
  + Containment of defects to specified parts of the system
  + Reactions to failures of a given severity
  + Self-monitoring of the operations and self-identification of defects Ability to allow specified work to continue after a failure of specified
  + severity for specified parts of the system under specified conditions
  + » Loss of specified operations (functionality requirement or set of functionality requirements) in case of failure of specified severities in specified periods of time for specified parts of the system
  + Loss of specified data in case of failure of specified severities in specified periods of time for specified parts of the system
* Robustness testing can, like the other technical tests, start at the requirements level, with testers reviewing robustness requirements.

**Recoverability Testing:**

* Recoverability is the product’s ability to reestablish its required level of performance and recover the data directly affected after a failure.
* This may cover aspects like:
  + Downtime after a failure of specified severities in specified periods of time for specified parts of the system
  + Uptime during specified periods of time for specified parts of the system over a specified period of time
  + Downtime during specified periods of time for specified parts of the system over a specified period of time
  + Time to reestablish consistent data in case of failure causing inconsistent data
  + Built-in backup facilities
  + Need for duplication (standby machine)
  + Redundancy in the software system
  + Reporting of effects of a crash
  + “Advice” in connection with restart

**Efficiency Testing:**

* In efficiency testing we test the requirements or needs concerned with the product’s ability to provide appropriate performance, relative to the amount of resources used, under stated conditions.
* The ISO 9126 standard breaks the efficiency attributes down into the subattributes:
  + Time behavior (performance)
  + Resource utilization
* **Performance Testing:**Time behavior or performance consists of the expectations towards the product’s ability to provide appropriate response and processing time and throughput rates when performing its functions under stated conditions.
* ***Load Testing:***Load testing is a special subtype of performance testing concerned with the product’s behavior under specified load conditions.
* **Stress Testing***:* Stress is an expression of the product’s capability for handling extreme situations
* ***Scalability Testing:***Scalability is the product’s ability to meet future efficiency requirements.

**Maintainability Testing:**

* In maintainability testing we test the requirements or needs concerned with the product’s ability to be analyzed and modified.
* Modifications may include corrections of defects, improvements or adaptations of the software to changes in the environment, and enhancements in requirements and functional specifications.
* The ISO 9126 standard breaks the efficiency attributes down into the subattributes:
  + Analyzability
  + Changeability
  + Stability
  + Testability
* Maintainability testing can be performed as static testing, where the structure, complexity, and other attributes of the code and the documentation are reviewed or undergoing inspections based on the pertaining maintenance requirements

**Analyzability Testing:**

* Analyzability is the ability of maintainers to identify deficiencies, diagnose the cause of failures, and identify areas requiring modification to implement required changes.
* Analyzability may cover aspects like:
  + Understandability: Making the design documentation, including the source code, understood by maintainers
  + Design standard compliance: Adherence to defined design standards
  + Coding standard compliance: Adherence to defined coding standards
  + Diagnosability: Presence and nature of diagnostic functions in the code
  + Traceability: Presence of traces between elements, for example, between requirements and test cases, and requirements and design and code
  + Technical manual helpfulness: Nature of any technical manual or specification

**Changeability Testing:**

* Changeability is the capability for implementation of a specified modification in the product
* Changeability may cover aspects like:
  + Modularity: The structure of the software
  + Code change efficiency: Capability for implementing required changes
  + Documentation change efficiency: Capability for documenting
  + implemented changes

**Stability Testing:**

* Stability is the capability of the product to avoid unexpected effects from modifications of the software, that is, to the risk of unexpected effects from modifications.
  + Data cohesion: Usage of data structures
  + Refailure rate: Pattern of new failures introduced as an effect of
  + implementation of required changes

**Testability Testing**

* Testability is the capability of validating the modified system, that is, how easy it is to perform testing of changes, either new tests or confirmation test, and how easy it is to perform regression testing.

**Portability Testing:**

* In portability testing we test the requirements or needs concerned with the product’s ability to be transferred into its intended environment.
* The environment may include the organization in which the product is used and the hardware, software, and network environment.
* The ISO 9126 standard breaks the portability attributes down into the subattributes:
  + Installability
  + Coexistence
  + Adaptability
  + Replaceability

**Installability Testing:**

* Installability is the capability of installing the product in a specified environment
  + *Space demand: Temporary space to be used during installation of* the software in a specified environment.
  + *Checking prerequisites: Facilities to ensure that the target environment* is meeting the demands of the product,
  + *Installation procedures: Existence and understandability of installation* aids such as general or specific installation scripts, installation manuals, or wizards. This may also include requirements concerning the time and effort to be spent on the installation task.
  + *Completeness: Facilities for checking that an installation is complete*
  + *Installation interruption: Possibility of interrupting an installation.* and rolling any work done back to leave the environment unchanged
  + *Customization: The capability of setting or changing parameters at*
  + installation time in a specified environment.
  + *Initialization: The capability of setting up initial information at installation*
  + time, both internal and external in a specified environment.
  + *Deinstallation: Facilities for removing the product partly (downgrading)*
  + or completely from the environment.

**Coexistence testing**:

* Coexistence is the software product’s capability to coexist with other independent software products in a common environment sharing common resources
* Coexistence can be very difficult to specify, since we don’t always know into which environment our software product is being placed.
* Coexistence testing can also be very difficult to perform, since it is usually impossible to establish correct test environments for this.

**Adaptability:**

* Adaptability is the capability of the software product to be adapted to different specified environments without applying actions or means other than those provided for this purpose for the system
* Adaptability may cover aspects like:
  + *Hardware dependency:*
  + *Software dependency:*
  + *Representation dependency:*
  + *Standard language conformance:*
  + *Dependency encapsulation:*
  + *Text convertability:*

**Replaceability Testing:**

* Replaceability is the capability of the product to be used in place of another specified product for the same purpose in the same environment.
* replaceability may cover aspects like
  + *Data loadability*
  + *Data convertability*