IF3141
SISTEM INFORMASI
System Testing

Semester I 2023/2024

1

System Testing

- System testing is conducted on a complete, integrated system (including software and hardware) to evaluate the system's compliance with its specified requirements.
- System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic

2

System Testing

- System testing takes, as its input, all of the integrated software components that have successfully passed integration testing.
- System testing is the final testing phase before user acceptance testing.

3

System Testing

- System testing have almost a destructive attitude and test not only the design, but also the behavior and even the believed expectations of the customer.
- It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specifications.

4

Types of System Testing

- Performance testing
- Load testing
- Stress testing
- Usability testing
- Security testing

5

Performance Testing

- The goal of performance testing is not to find bugs, but to eliminate bottlenecks and establish a baseline for future regression testing.
- To conduct performance testing is to engage in a carefully controlled process of measurement and analysis.
- Ideally, the software under test is already stable enough so that this process can proceed smoothly.

6

Set of Expectations

- A clearly defined set of expectations is essential for meaningful performance testing. If you don't know where you want to go in terms of the performance of the system, then it matters little which direction you take.
- For example, for a Web application, you need to know at least two things:
- Expected load in terms of concurrent users or HTTP connections.
- Acceptable response time..

7

Looking for Bottlenecks

- Once you know where you want to be, you can start on your way there by constantly increasing the load on the system while looking for bottlenecks.
- To take again the example of a Web application, these bottlenecks can exist at multiple levels, and to pinpoint them you can use a variety of tools.

8

Looking for Bottlenecks

- At the application level, developers can use profilers to spot inefficiencies in their code.
- At the database level, developers can use database-specific profilers and query optimizers.
- At the operating system level, system engineers can use utilities such as top, vmstat, iostat (on Unix-type systems) and PerfMon (on Windows) to monitor hardware resources such as CPU, memory, swap, disk I/O; specialized kernel monitoring software can also be used.

9

Looking for Bottlenecks

- At the network level, network engineers can use packet sniffers such as tcpdump, network protocol analyzers such as ethereal, and various utilities such as netstat, MRTG, ntop, mii-tool.

10

Performance Tuning

- When the results of the load test indicate that performance of the system does not meet its expected goals, it is time for tuning, starting with the application and the database.
- You want to make sure your code runs as efficiently as possible and your database is optimized on a given OS/hardware configurations.

11

Performance Tuning

- If, after tuning the application and the database, the system still doesn't meet its expected goals in terms of performance, a wide array of tuning procedures is available at the all the levels discussed before.
- Here are some examples of things you can do to enhance the performance of a Web application outside of the application code per se:

12

Performance Tuning

- Use Web cache mechanisms, such as the one provided by Squid.
- Publish highly-requested Web pages statically, so that they don't hit the database.
- Scale the Web server farm horizontally via load balancing.
- Scale the database servers horizontally and split them into read/write servers and read-only servers, then load balance the read-only servers.
- Scale the Web and database servers vertically, by adding more hardware resources (CPU, RAM, disks).
- Increase the available network bandwidth.

13

One Variable at a Time

- Performance tuning can sometimes be more art than science, due to the sheer complexity of the systems involved in a modern Web application. Care must be taken to modify one variable at a time and redo the measurements, otherwise multiple changes can have subtle interactions that are hard to qualify and repeat.

14

Staging Environments

- In a standard test environment such as a test lab, it will not always be possible to replicate the production server configuration. In such cases, a staging environment is used which is a subset of the production environment. The expected performance of the system needs to be scaled down accordingly.

15

Baseline

- The cycle "run load test -> measure performance -> tune system" is repeated until the system under test achieves the expected levels of performance.
- At this point, testers have a baseline for how the system behaves under normal conditions.
- This baseline can then be used in regression tests to gauge how well a new version of the software performs.

16

Benchmarks

- Another common goal of performance testing is to establish benchmark numbers for the system under test.
- There are many industry-standard benchmarks such as the ones published by TPC, and many hardware/software vendors will fine-tune their systems in such ways as to obtain a high ranking in the TCP top-tens.

17

Load Testing

- In the testing literature, the term "load testing" is usually defined as the process of exercising the system under test by feeding it the largest tasks it can operate with.
- Load testing is sometimes called volume testing, or longevity/endurance testing.

18

Examples of Volume Testing

- Testing a word processor by editing a very large document.
- Testing a printer by sending it a very large job.
- Testing a mail server with thousands of users mailboxes.
- A specific case of volume testing is zero-volume testing, where the system is fed empty tasks.

19

Examples of Longevity/Endurance Testing

- Testing a client-server application by running the client in a loop against the server over an extended period of time.

20

Goals of Load Testing

- Expose bugs that do not surface in cursory testing, such as memory management bugs, memory leaks, buffer overflows, etc.
- Ensure that the application meets the performance baseline established during performance testing.
- This is done by running regression tests against the application at a specified maximum load.

21

Performance vs. Load Testing

- Performance testing uses load testing techniques and tools for measurement and benchmarking purposes and uses various load levels.
- Load testing operates at a predefined load level, usually the highest load that the system can accept while still functioning properly.
- Note that load testing does not aim to break the system by overwhelming it, but instead tries to keep the system constantly humming like a well-oiled machine.

22

Large Datasets

- In the context of load testing, there is extreme importance of having large datasets available for testing.
- Many important bugs simply do not surface unless you deal with very large entities: thousands of users in repositories such as LDAP/NIS/Active Directory, thousands of mail server mailboxes, multi-gigabyte tables in databases, deep file/directory hierarchies on file systems, etc.

23

Stress Testing

- Stress testing tries to break the system under test by overwhelming its resources or by taking resources away from it (in which case it is sometimes called negative testing).
- The main purpose behind this madness is to make sure that the system fails and recovers gracefully -- this quality is known as recoverability.
- Where performance testing demands a controlled environment and repeatable measurements, stress testing joyfully induces chaos and unpredictability

24

Examples of Stress Testing

- Double the baseline number for concurrent users/HTTP connections.
- Randomly shut down and restart ports on the network switches/routers that connect the servers (via SNMP commands for example).
- Take the database offline, then restart it.
- Rebuild a RAID array while the system is running.
- Run processes that consume resources (CPU, memory, disk, network) on the Web and database servers.

25

Goals of Stress Testing

- Stress testing does not break the system purely for the pleasure of breaking it, but instead it allows testers to observe how the system reacts to failure.
- Does it save its state, or does it crash suddenly?
- Does it just hang and freeze, or does it fail gracefully?

26

Goals of Stress Testing

- On restart, is it able to recover from the last good state?
- Does it print out meaningful error messages to the user, or does it merely display incomprehensible hex codes?
- Is the security of the system compromised because of unexpected failures?
- And the list goes on.

27

Usability Testing

- Usability testing is a technique used in design. It is a means for the designer to improve the product by measuring its usability, or user-friendliness.
- The product is some human-made object (such as a web page, a computer interface, a document, or a device) for its intended purpose.

28

Goals of Usability Testing

- If usability testing uncovers difficulties, such as people having difficulty understanding instructions, manipulating parts, or interpreting feedback, then developers should improve the design and test it again.
- During usability testing, the aim is to observe people using the product in as realistic a situation as possible, to discover errors and areas of improvement.

29

Usability Metrics

- Usability testing generally involves measuring how well test subjects respond in four areas: time, accuracy, recall, and emotional response.
- The results of the first test can be treated as a baseline or control measurement; all subsequent tests can then be compared to the baseline to indicate improvement.

30

Usability Metrics

- Time on Task: How long does it take people to complete basic tasks? (For example, find something to buy, create a new account, and order the item.)
- Accuracy: How many mistakes did people make? (And were they fatal or recoverable with the right information?)
- Recall: How much does the person remember afterwards or after periods of non-use?
- Emotional Response: How does the person feel about the tasks completed? (Confident? Stressed? Would the user recommend this system to a friend?)

31

What Usability Testing is Not

- Simply gathering opinions on an object or document is market research rather than usability testing.
- Usability testing usually involves a controlled experiment to determine how well people can use the product.

32

What Usability Testing is Not

- Rather than showing users a rough draft and asking, "Do you understand this?", usability testing involves watching people trying to use something for its intended purpose.
- For example, when testing instructions for assembling a toy, the test subjects should be given the instructions and a box of parts. Instruction phrasing, illustration quality, and the toy's design all affect the assembly process.

33

Methods of Usability Testing

- Setting up a usability test involves carefully creating a scenario, or realistic situation, wherein the person performs a list of tasks using the product being tested while observers watch and take notes.
- Several other test instruments such as scripted instructions, paper prototypes, and pre- and post-test questionnaires are also used to gather feedback on the product being tested.

34

Methods of Usability Testing

- For example, to test the attachment function of an e-mail program, a scenario would describe a situation where a person needs to send an e-mail attachment and ask him or her to undertake this task.
- The aim is to observe how people function in a realistic manner, so that developers can see problem areas, and what people like.

35

Methods of Usability Testing

- Techniques popularly used to gather data during a usability test include think aloud protocol and eye tracking.

36

Think Aloud Protocols

- Think aloud protocols involve participants thinking aloud as they are performing a set of specified tasks.
- Users are asked to say whatever they are looking at, thinking, doing, and feeling, as they go about their task.
- This enables observers to see first-hand the process of task completion (rather than only its final product).

37

Think Aloud Protocols

- Observers at such a test are asked to objectively take notes of everything that users say, without attempting to interpret their actions and words.
- Test sessions are often audio and video taped so that developers can go back and refer to what participants did, and how they reacted.
- The purpose of this method is to make explicit what is implicitly present in subjects who are able to perform a specific task.

38

Eye Tracking

- Eye tracking is the process of measuring either the point of gaze ("where we are looking") or the motion of an eye relative to the head.
- An eye tracker is a device for measuring eye positions and eye movements.
- The most widely used current designs are video-based eye trackers. A camera focuses on one or both eyes and records their movement as the viewer looks at some kind of stimulus.

39

Hallway Usability Testing

- Hallway usability testing is a specific methodology of software usability testing.
- Rather than using an in-house, trained group of testers, just five to six random people are brought in to test the software.
- The theory, as adopted from Jakob Nielsen's research, is that 95% of usability problems can be discovered using this technique.

40

Hallway Usability Testing

- In the early 1990s, Jakob Nielsen popularized the concept of using numerous small usability tests -- typically with only five test subjects each -- at various stages of the development process.
- His argument is that, once it is found that two or three people are totally confused by the home page, little is gained by watching more people suffer through the same flawed design.

41

Hallway Usability Testing

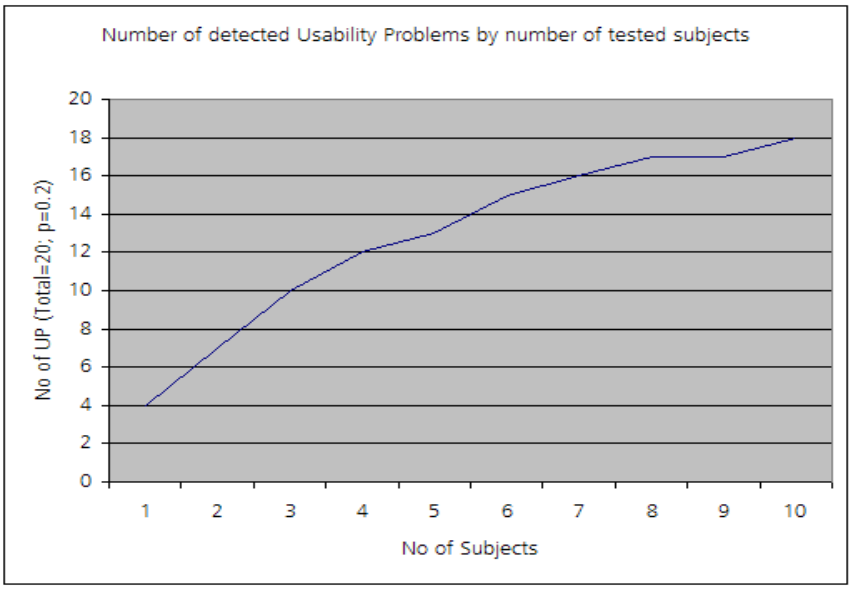
- The claim of "Five users is enough" was later described by a mathematical model which states for the proportion of uncovered problems U

  $U = 1 - (1 - p)n$

  where p is the probability of one subject identifying a specific problem and n is the number of subjects.
- This model shows up as an asymptotic graph towards the number of real existing problems.

42

Hallway Usability Testing



43

Two Key Challenges

- First, since usability is related to the specific set of users, such a small sample size is unlikely to be representative of the total population so the data from such a small sample is more likely to reflect the sample group than the population they may represent.
- Second, many usability problems encountered in testing are likely to prevent exposure of other usability problems, making it impossible to predict the percentage of problems that can be uncovered without knowing the relationship between existing problems.

44

Hallway Usability Testing

- Most researchers today agree that, although 5 users can generate a significant amount of data at any given point in the development cycle, in many applications a sample size larger than five is required to detect a satisfying amount of usability problems.
- These users can be divided into small tests with five users each.

45

Security Testing

- Security testing is the process to determine that a system protects data and maintains functionality as intended.
- The six basic security concepts that need to be covered by security testing are: confidentiality, integrity, authentication, authorization, availability and non-repudiation.

46

Confidentiality

- A security measure which protects against the disclosure of information to parties other than the intended recipient that is by no means the only way of ensuring confidentiality.

47

Integrity

- A measure intended to allow the receiver to determine that the information which it receives has not been altered in transit or by other than the originator of the information.
- Integrity schemes often use some of the same underlying technologies as confidentiality schemes, but they usually involve adding additional information to a communication to form the basis of an algorithmic check rather than the encoding all of the communication.

48

Authentication

- A measure designed to establish the validity of a transmission, message, or originator.
- Allows a receiver to have confidence that information it receives originated from a specific known source.

49

Authentication

- A brute force attack is an automated process of trial and error used to guess a person's username, password, credit-card number or cryptographic key.
- Insufficient authentication occurs when a web site permits an attacker to access sensitive content or functionality without having to properly authenticate.
- Weak password recovery validation is when a web site permits an attacker to illegally obtain, change or recover another user's password.

50

Authentication

- Credential/Session Prediction is a method of hijacking or impersonating a web site user.
- Insufficient Authorization is when a web site permits access to sensitive content or functionality that should require increased access control restrictions.

51

Authentication

- Insufficient Session Expiration is when a web site permits an attacker to reuse old session credentials or session IDs for authorization.
- Session Fixation is an attack technique that forces a user's session ID to an explicit value.

52

Availability

- Assuring information and communications services will be ready for use when expected.
- Information must be kept available to authorized persons when they need it.

53

Non-Repudiation

- A measure intended to prevent the later denial that an action happened, or a communication that took place etc.
- In communication terms this often involves the interchange of authentication information combined with some form of provable time stamp.

54