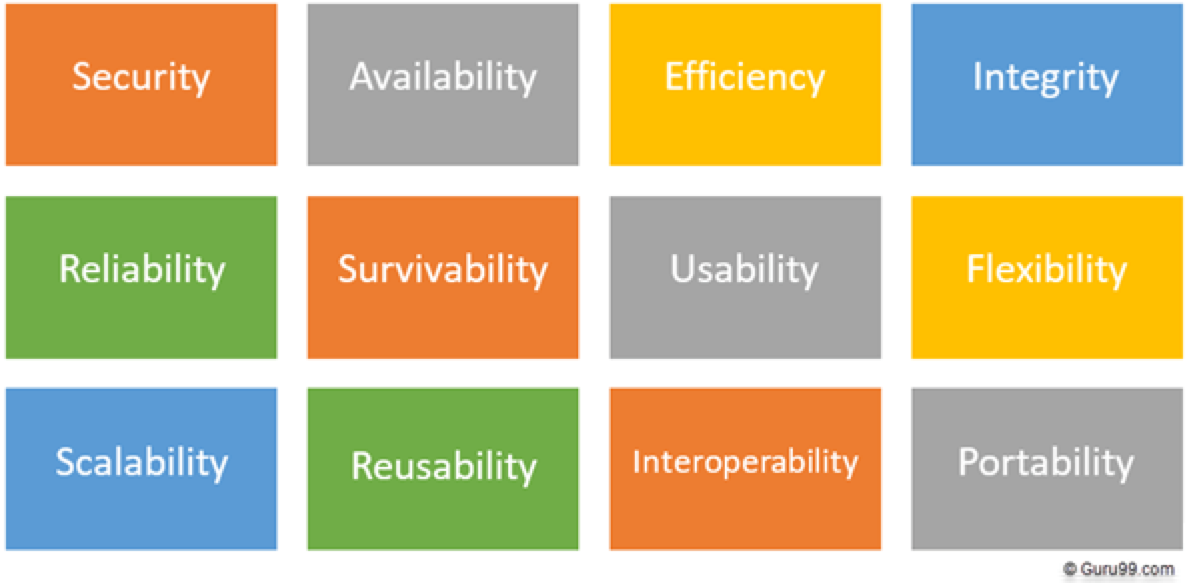
EN.601.422 / EN.601.622

Software Testing & Debugging

Non-functional Testing



Software Testing

- **Functional Testing:** verifies the operational execution of a software
    - unit, integration, and system testing
  - typically starts before non-functional testing
- **Non-functional Testing:** tests aspects of the software other than its functionality
  - making sure interests of the end-user are respected
  - **vital to add market value to the product**
  - is a blackbox testing

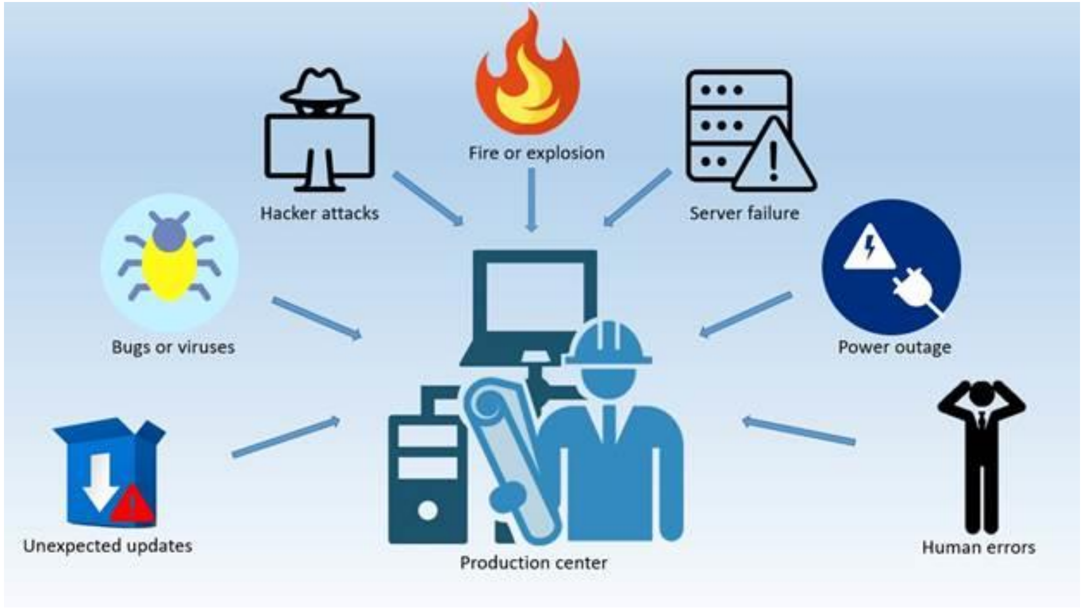
** Non-functional testing must be measurable; there is no place for subjective characterizations like good, better, best, etc. **

Load Testing

How does the application behave when too many users access it concurrently?

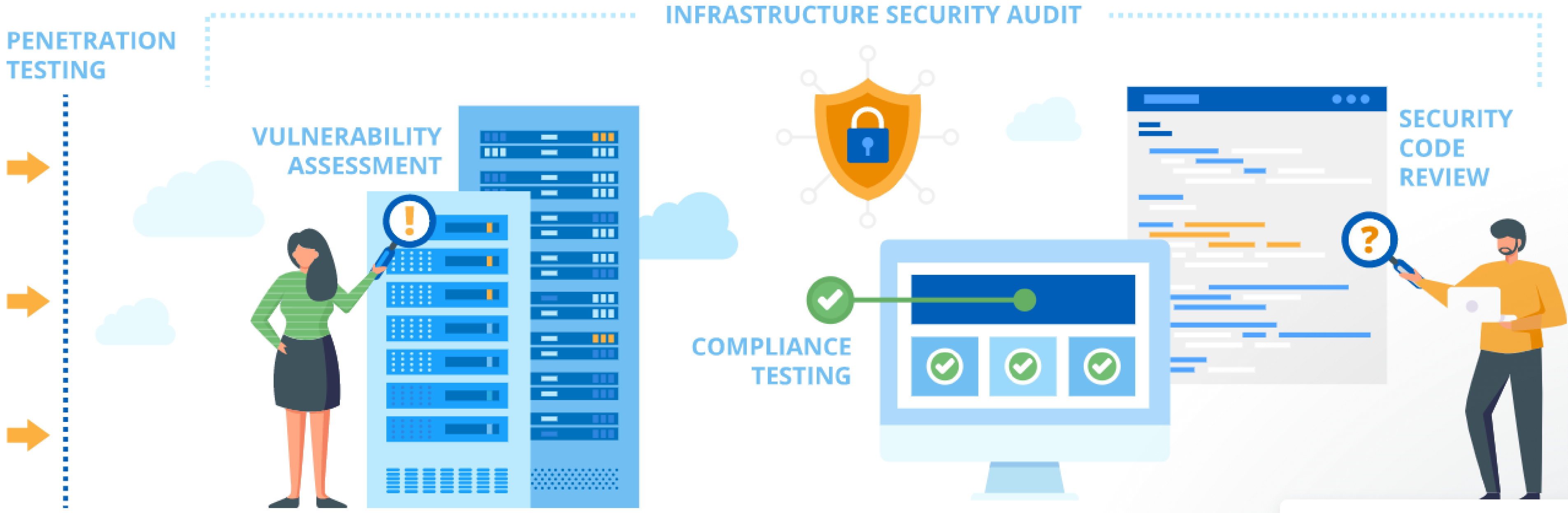
Recovery Testing

Can the application recover from a crash or a failure?



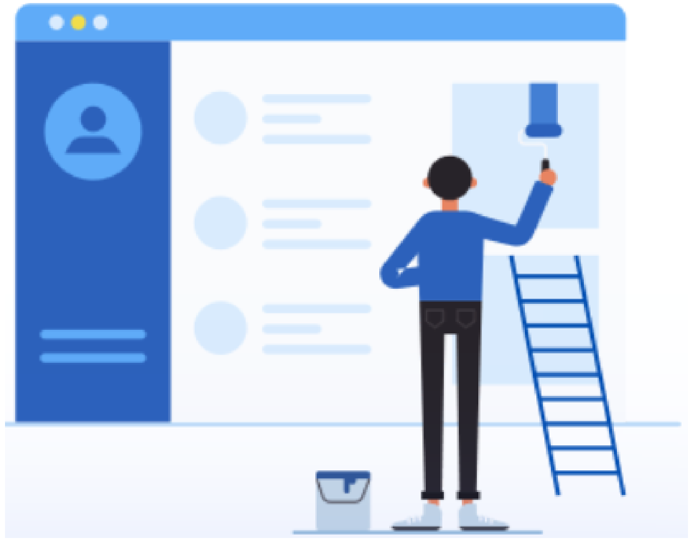
Security Testing

How secure is the application?



Usability Testing

Is the user interface of the application intuitive and user-friendly enough?

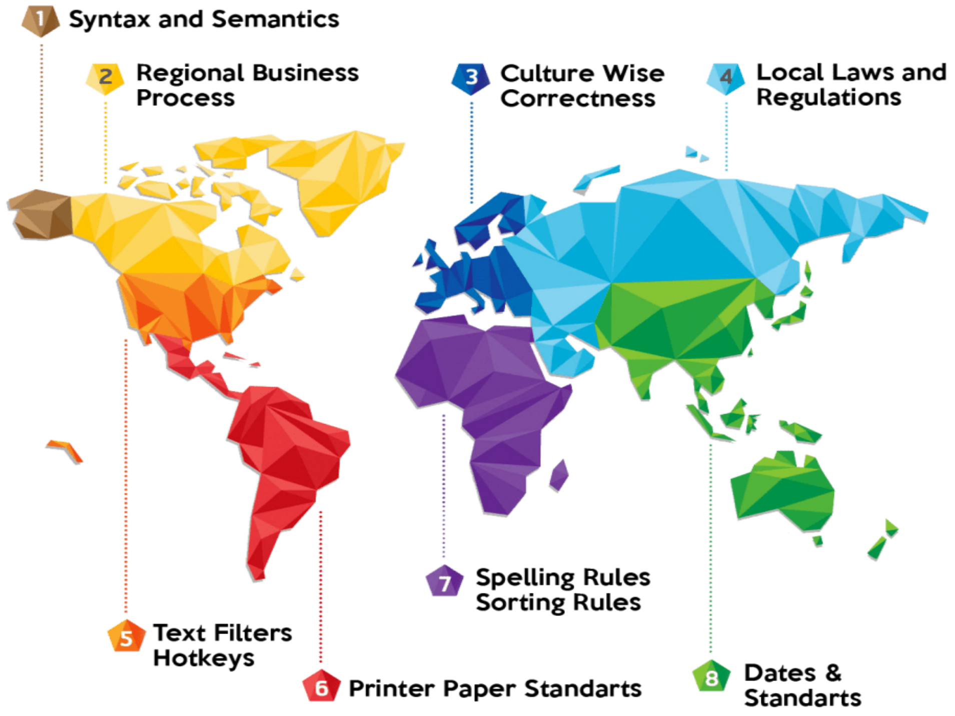


Accessibility Testing

Can disabled people use the application too?

Internationalization/localization Testing

Can people with different languages and regional peculiarities use the application?


1 Syntax and Semantics
2 Regional Business Process
3 Culture Wise Correctness
4 Local Laws and Regulations
5 Text Filters Hotkeys
6 Printer Paper Standarts
7 Spelling Rules Sorting Rules
8 Dates & Standarts

Documentation Testing

Are the documents/user manual provided with the application easy to understand?

Software Requirements

- **Non-functional** (aka quality) requirements can be as important as functional requirements, if not more important!
  - security of a banking app
  - user friendliness of a web app for kids or tech illiterate
  - responsiveness of an app with lots of concurrent users
  - Etc.
- Can use the same “user story” cards for non-functional requirements

Performance Testing

“Determine how a system performs in terms of responsiveness and stability under a particular workload/conditions.”

- Kinds of Performance Testing:
  - **Load Testing**
  - Stress Testing
  - Spike Testing
  - Soak Testing
  - Configuration Testing

https://en.wikipedia.org/wiki/Software_performance_testing

Load Testing

"Modeling the ***expected usage*** of a software ***by simulating multiple users*** accessing the software ***concurrently***"

GOAL: improve *performance bottlenecks* and ensure smooth functioning

Examples:

- **Printer**: Testing a printer by transferring a large number of documents for printing concurrently
- **Hard Disk**: Concurrent transfer of multiple files to and from the disk
- **A government web portal**: thousands/millions concurrent users

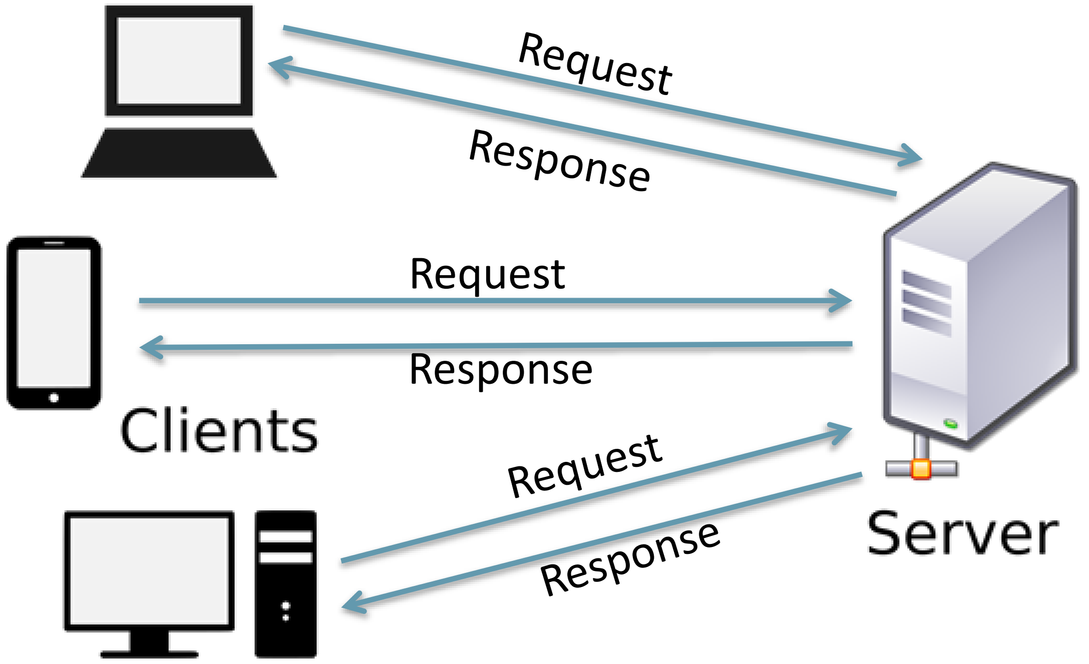
Importance of Load Testing

"According to a survey, ***75% users*** said that if a ***site crashed*** or if it was slow, they would ***leave the site***"

"***50% of the users*** said that they will shop elsewhere if the website or app did ***not load in 3 seconds***"

http://tryqa.com/what-is-load-testing-in-software/

Client-Server



Client-Server

- The <u>**Client**</u> is the application that runs on the end-user computer that sends “**requests**” to the server.
  - E.g., browser

- The <u>**Server**</u> is the application that receives requests from the clients and contains the logic to send the appropriate “**response**” back to the client. The server usually has an application programming interface (API), and often includes a database, which will persistently store the data for the application

Load Testing a Website

Load Testing typically involves **simulating the load!**

- Steps:
  1. creating a pertinent scenario,
  2. scripting the scenario as a test,
  3. creation of virtual users,
  4. having the (virtual) users replay the test,
  5. analyzing test results

Load Testing Metrics

- Response Metrics
  - Average Response Time
  - Peak Response Time (PRS)
  - Error Rate
- Volume Metrics
  - Concurrent Users
  - Request Per Second (RPS)
  - Throughput

Response Metrics

Response Time Metrics

Acceptable Response Time Range

- **Average Response Time**: Mean of every roundtrip request/response cycle
- **Peak Response Time**: longest response time
- **Error Rate**: percentage of problem requests compared to all requests.

- **0.1 second**: “Instant” Response
- **1.0 second**: Acceptable, but not ideal
- **5+ seconds**: User will avoid website or app

Volume Metrics

- **Concurrent Users:** number of (virtual) users active at any point in time.
- **Request Per Second:** Number of requests being sent to the target server in 1 second
- **Throughput:** a measurement of bandwidth consumed during the test typically calculated as:

$$\frac{\textit{kilobytes}}{\textit{time}}, \textit{ or}$$

$$\frac{\textit{\# of requests}}{\textit{time}}$$

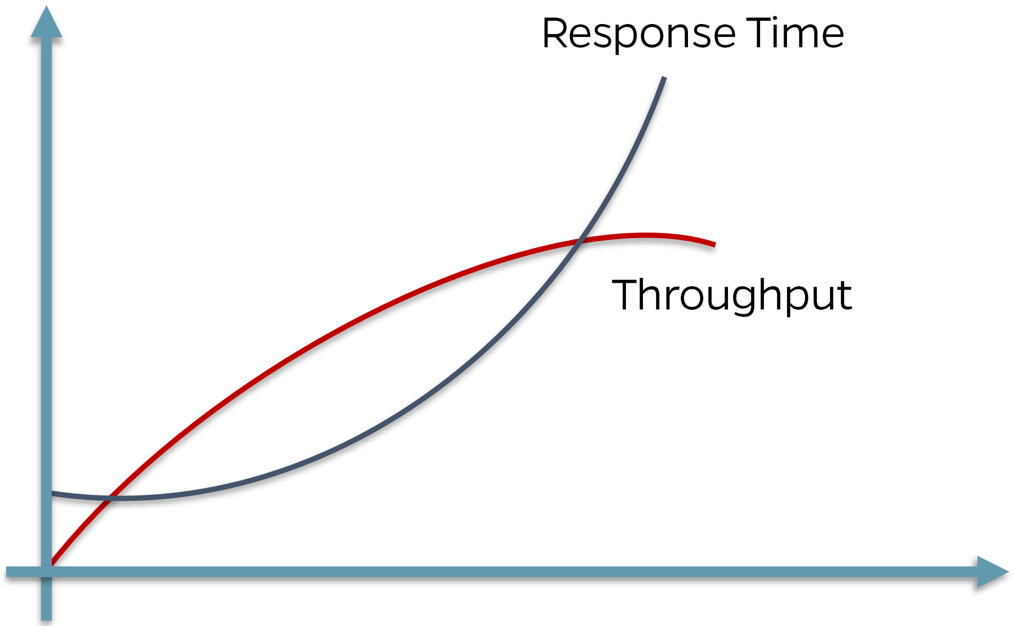
Throughput

JOE'S GAS
JOE'S GAS
JOE'S GAS

3 Trucks
3 Gas Pumps

Throughput = 3 / minute
(no waiting)

Response Time vs. Throughput Trend



Tools

- Commercial:
  - LoadNinja, WebLOAD, LoadView, StresStimulus, etc.
- Open Source:
  - **Apache JMeter**, Locust, Gatling, etc.

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING