

Principles of Agile Testing

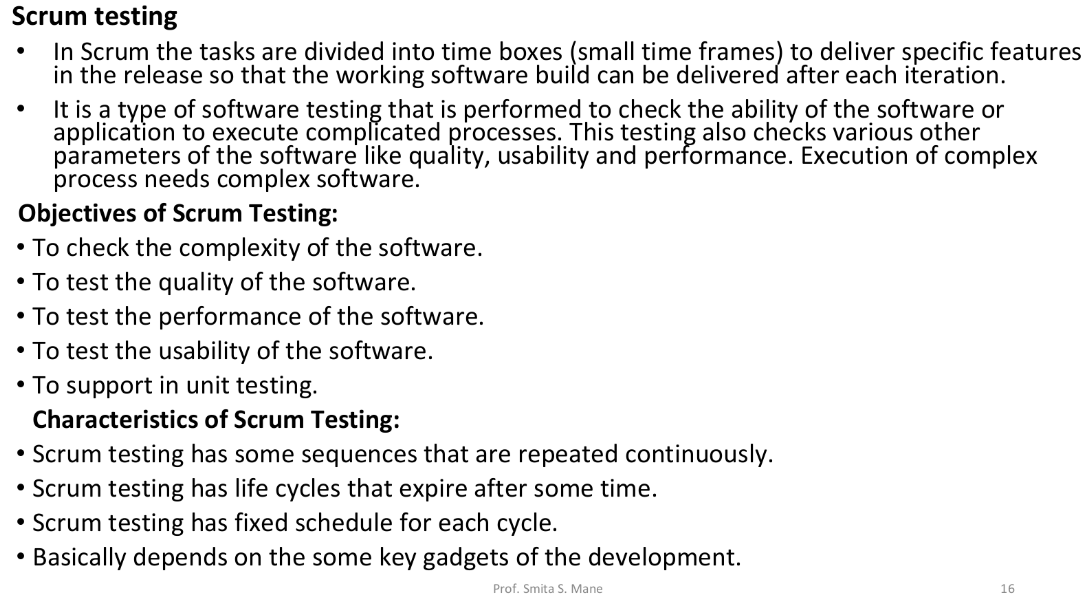
1. Testing moves the project forward
2. Testing is NOT a phase
3. Everyone tests
4. Shortening feedback loops
5. Keep the code clean
6. Done done, not just done
7. Lightwieght documentation
8. Test driven development

Agile Test Cycle

1. Impact Assessment
2. Agile Test Planning
3. Release readiness
4. Daily scrums
5. Test agility review

Challenges in Agile Testing

1. Changes in requirements
2. Not enough communication
3. Skilled testers
4. Not enough data
5. Testing continuously
6. Frequent Regression testing



| **Aspect** | **Traditional Software** | **Web-Based Software** |
| --- | --- | --- |
| Deployment | Installed on individual devices | Accessed via web browsers |
| Accessibility | Limited to specific devices | Accessible from any device with an internet connection |
| Installation & Updates | Manual installation and updates | Updates are typically automatic and seamless |
| Cost | May require a one-time purchase | Often subscription-based, pay-as-you-go |
| Local Storage | Uses local device storage | Data stored on remote servers (cloud) |
| Data Security | Data stored locally, requires local backup | Data stored remotely, may have robust security measures |
| Maintenance | Requires manual maintenance | Centralized maintenance by the service provider |
| Collaboration | Limited real-time collaboration | Real-time collaboration and sharing features |
| Offline Access | Generally requires internet access | Some web apps offer limited offline functionality |
| Compatibility | Platform-specific (e.g., Windows, Mac) | Cross-platform compatibility (works on multiple OS) |
| Customization | Customization options may be limited | Often offers flexibility and customization through settings |
| Scalability | Limited scalability without additional licenses | Scalable based on subscription or usage needs |
| User Data | User data remains on local devices | User data can be accessed from any device |
| Accessibility | Limited accessibility options | Web accessibility standards may be applied |
| Support and Help | May require contacting support for issues | Web-based support and community forums often available |
| Software Size | Larger installation files | Smaller as most resources are fetched online |
| Integration | May require third-party plugins for integration | Often supports integration with other web services |

Evaluation of web technology:

1. \*\*Reliability:\*\*

- Web technology should be dependable and consistent. Evaluate it by examining uptime and downtime records, as well as error rates.

- Assess how the technology handles unexpected events, such as server crashes, and whether it has backup and recovery mechanisms in place.

- Consider the technology's history of updates and bug fixes to gauge its reliability.

2. \*\*Performance:\*\*

- Analyze the response time and load times of web applications or websites. Faster load times often result in a better user experience.

- Evaluate the technology's ability to handle high traffic loads without degrading performance.

- Use performance testing tools to measure the system's capacity and identify potential bottlenecks.

3. \*\*Security:\*\*

- Security is paramount for web technology. Evaluate it in terms of data protection, user authentication, and access control.

- Look for regular security updates and a clear security policy from the provider.

- Consider third-party security audits and penetration testing to identify vulnerabilities.

4. \*\*Usability:\*\*

- Usability refers to how easily users can interact with the technology. Assess it through user testing, surveys, and feedback.

- Evaluate the user interface, user experience, and accessibility features.

- Ensure that the technology complies with web accessibility standards (e.g., WCAG) for a broader user base.

5. \*\*Scalability:\*\*

- Examine the ability of the web technology to handle growth. Can it accommodate increased data, users, and traffic?

- Consider the ease of scaling, whether vertically (adding more resources to a single server) or horizontally (adding more servers to distribute the load).

- Assess the cost implications of scaling the technology.

6. \*\*Availability:\*\*

- High availability is essential to minimize downtime. Evaluate the technology's uptime guarantee and track record.

- Consider redundancy and failover mechanisms in place, such as load balancing and backup servers.

- Analyze the provider's disaster recovery and backup strategies.

7. \*\*Maintainability:\*\*

- This pertains to the ease of maintaining and updating the technology. Assess how updates and maintenance are handled.

- Look for version control, automated deployment, and rollback mechanisms.

- Evaluate the community and support for the technology, as this can affect maintainability.

How to test web-based systems?

1. Functionality testing - features
2. Usability testing - alt for image, ease of navigation, test content(grammatical and spelling)
3. Interface testing - application, web server, database server
4. Database testing
5. Compatibility testing - browser compatibility
6. Performance testing
7. Security testing
8. Crowd testing