Quality Resources & Solutions

ISTQB FOUNDATION LEVEL

Tài liệu lý thuyết

Giảng viên: Tạ Thị Thinh

Email: thinhtt0204@gmail.com

SDT: 0986775464

Skype: ta.thinh0204

Chapter 0: ISTQB introduction

Nội dung chính:

1. **ISTQB Foundation Level Examination**

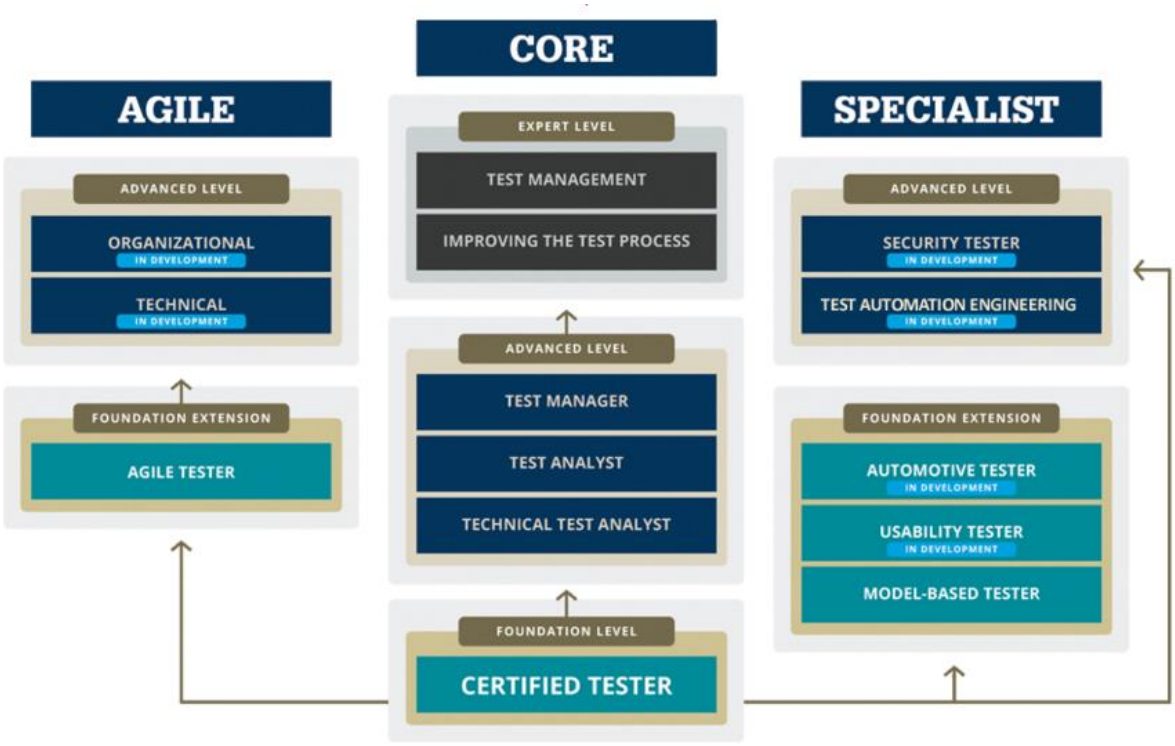
- Giới thiệu về bài thi ISTQB
- Who is ISTQB -ISTQB là gì?
- Course structure -Exams in ISTQB
- About Foundation level Examination -Về các mức độ kiến thức trong bài thi

2. **How to study ISTQB and prepare for exam**

- Plan for examination- Lập kế hoạch cho kỳ thi
- How to read and use ISTQB books- Cách đọc và sử dụng các cuốn sách ISTQB bằng tiếng Anh
- Tips for doing exercises-Các chỉ dẫn, kinh nghiệm để làm bài tập

1. ISTQB Foundation Level Examination

1.1 ISTQB Course structure (Cấu trúc ISTQB)



- There are currently three levels of certification

Có 3 mức độ của chứng chỉ

  - The Foundation Level certification (one module) - CTFL (Certified Tester Foundation level

Chứng chỉ ở mức cơ bản (1 mảng)

  - Advanced Level certifications (three modules)

Chứng chỉ ở cao hơn (3 mảng: quản lý, kỹ thuật test và automation)

  - An Expert levels is currently being developed

Chứng chỉ ở mức chuyên gia đang được xây dựng

- For all three levels, international working groups develop and maintain internationally uniform curricula and exams - Đối với tất cả 3 mức chứng chỉ, thì đều thống nhất tài liệu giảng và bài thi trên toàn thế giới.

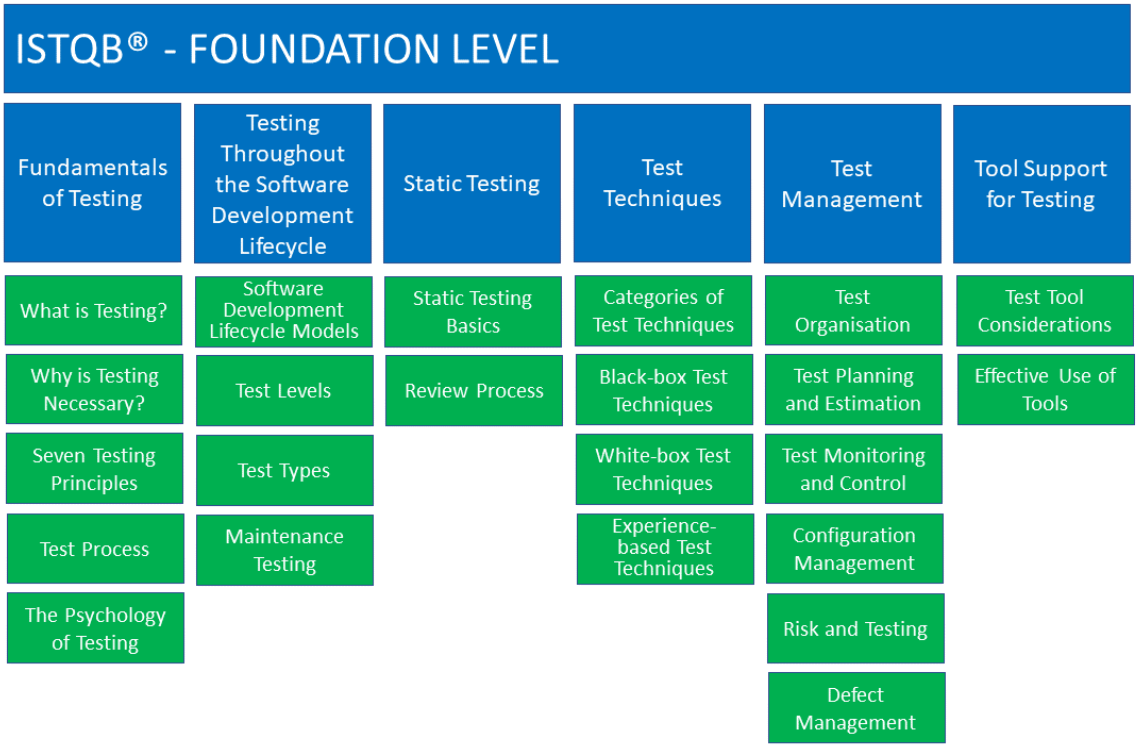
1.2 Who is ISTQB (ISTQB là gì)?

- ISTQB là 1 tổ chức phi lợi nhuận có trách nhiệm định nghĩa ra các hướng dẫn như cấu trúc bài thi, quy ước, khái niệm, chứng chỉ.
- Nhóm làm việc trong tổ chức ISTQB có trách nhiệm phát triển và duy trì các giáo trình và xây dựng bài thi chung trên toàn thế giới.

1.3 Exams in ISTQB (Bài thi trong ISTQB)

- The ISTQB Certified Tester program provides certification for software testers internationally.

Chương trình thi chứng chỉ ISTQB cho tester cung cấp chứng chỉ về test phần mềm có giá trị trên toàn thế giới.



2. How to study ISTQB and prepare for exam

Cách nghiên cứu và chuẩn bị cho kỳ thi lấy chứng chỉ ISTQB

2.1 How to read and use ISTQB books

Cách đọc và sử dụng các cuốn sách ISTQB bằng tiếng Anh

Sách dùng để thi chính thức ISTQB gồm có:

1. ISTQB FOUNDATIONS LEVEL

Giới thiệu và giải thích 1 cách đầy đủ về các khái niệm và hoạt động của testing. Gồm có 207 trang.

2. ISTQB FOUNDATION LEVEL SYLLABUS

Tóm tắt các khái niệm chính về testing, những key word cần nhớ của quyển 1
3. ISTQB GLOSSARY OF TESTING TERMS
Từ điển các khái niệm về testing
Ngoài ra các bộ đề thi, các ví dụ về mẫu bài thi có rất nhiều trên mạng

Mục tiêu nghiên cứu các tài liệu ISTQB:

- K1 Remember – Ghi nhớ
- K2 Understand – Hiểu biết
- K3 Apply – Áp dụng

2.2 Foundation Level examination – Bài thi ở mức cơ sở

- 40 multiple choice questions

40 câu hỏi trắc nghiệm

- A scoring of I point for each correct answer, each question can be 1 to 2 correct answers

Một đáp án đúng được 1 điểm, mỗi câu hỏi có thể có 1-2 câu trả lời đúng

- A pass mark of 65% (26 or more points)

Điểm đậu là 65% (26 câu đúng hoặc nhiều hơn)

- A duration of 60 minutes (or 75 minutes for candidates taking exams that are not in their native or local language) - Thời gian làm bài 60 phút (hoặc 75 phút cho các ứng cử viên tham gia các kỳ thi mà không phải là ngôn ngữ mẹ đẻ hoặc của địa phương)

Question details – phân chia câu hỏi

- K1 – 8 câu hỏi
- K2 – 24 câu hỏi
- K3 – 8 câu hỏi

2.3 Plan for examination -Lập kế hoạch cho kỳ thi

- Chuẩn bị thi trong vòng 1 tháng đến 1.5 tháng
- Mỗi tuần tối thiểu phải đọc hết 1 chương và làm bài tập

Tips for doing exercises -Các chỉ dẫn, kinh nghiệm để làm bài tập

- Đọc hiểu và kỹ lưỡng quyển Syllabus là cơ sở cho việc ôn luyện
- Kết hợp giữa vừa đọc sách vừa làm bộ đề để ghi nhớ
- Kết quả đúng không quan trọng bằng dẫn chứng chỉ ra trong sách để tìm được câu trả lời đúng
- Đừng cố gắng dịch sách để hiểu từng câu từng chữ trong sách, hãy đọc 1 lượt, 2 lượt, 3 lượt 1 chương và kết hợp làm bài tập để hiểu được ý nghĩa và ghi nhớ
- Ghi nhớ từ tiếng Anh và câu hỏi tiếng Anh vì bài thi bằng tiếng Anh
- Làm quen với câu hỏi dài.

Chapter 1: Fundamentals of Testing

1.1 What is Testing? - Testing là gì

What is testing?

- A common misperception of testing is that it only consists of running tests, i.e., executing the software. This is part of testing, but not all of the testing activities - Thường mọi người hiểu khái niệm test chỉ là chạy test, chạy phần mềm nhưng đó chỉ là 1 phần ko phải tất cả các hoạt động test
- Test activities exist before and after test execution. These activities include planning and control, choosing test conditions, designing and executing test cases, checking results, evaluating exit criteria, reporting on the testing process and system under test, and finalizing or completing closure activities after a test phase has been completed - Các hoạt động test tồn tại trước và sau khi chạy PM bao gồm: lên kế hoạch và kiểm soát, chọn điều kiện test, thiết kế và chạy test case, test kết quả, đánh giá tiêu chí kết thúc, báo cáo trong quy trình test và các hoạt động đóng sau khi giai đoạn test hoàn thành.
- Testing also includes reviewing documents (including source code) and conducting static analysis - Test thì bao gồm cả review tài liệu, source code, phân tích tĩnh
- Both dynamic testing and static testing can be used as a means for achieving similar objectives, and will provide information that can be used to improve both the system being tested and the development and testing processes - Cả dynamic và static testing được sử dụng đồng thời để đạt được mục đích giống nhau và sẽ cung cấp thông tin để cải tiến HT đang đc test và các quy trình

1.1.1 Typical Objectives of Testing

- To evaluate work products such as requirements, user stories, design, and code - Để đánh giá các sản phẩm như là requirements, user stories, design, and code
- To verify whether all specified requirements have been fulfilled - Để xác minh xem tất cả các yêu cầu đã được thực hiện đầy đủ chưa
- To validate whether the test object is complete and works as the users and other stakeholders expect - Để xác thực xem đối tượng test đã hoàn thành chưa và hoạt động như người dùng và các bên liên quan khác mong đợi chưa
- To build confidence in the level of quality of the test object - Để xây dựng sự tự tin về mức độ chất lượng
- To prevent defects - Để ngăn ngừa lỗi
- To find failures and defects - Để tìm failures và lỗi
- To provide sufficient information to stakeholders to allow them to make informed decisions, especially regarding the level of quality of the test object - Cung cấp đầy đủ thông tin cho các bên liên quan để cho phép họ đưa ra quyết định sáng suốt, đặc biệt là về mức độ chất lượng của đối tượng test
- To reduce the level of risk of inadequate software quality (e.g., previously undetected failures occurring in operation) - Để giảm mức độ rủi ro về chất lượng phần mềm không đầy đủ (ví dụ: các lỗi không được phát hiện trước đó xảy ra trong khi hoạt động)
- To comply with contractual, legal, or regulatory requirements or standards, and/or to verify the test object's compliance with such requirements or standards - Tuân thủ các yêu cầu hoặc tiêu chuẩn theo

hợp đồng, pháp lý hoặc theo quy định và / hoặc để xác minh đối tượng test tuân thủ các yêu cầu hoặc tiêu chuẩn đó

Different viewpoints in testing take different objectives into account

Có các mục đích khác nhau trong từng giai đoạn test:

- During component testing, one objective may be to find as many failures as possible so that the underlying defects are identified and fixed early. Another objective may be to increase code coverage of the component tests - Trong component testing, một mục tiêu có thể là tìm ra càng nhiều lỗi càng tốt để các defect cơ bản được xác định và khắc phục sớm. Một mục tiêu khác có thể là tăng độ bao phủ test của các thành phần.
- During acceptance testing, one objective may be to confirm that the system works as expected and satisfies requirements. Another objective of this testing may be to give information to stakeholders about the risk of releasing the system at a given time - Trong quá trình test chấp nhận, một mục tiêu có thể là xác nhận rằng hệ thống hoạt động như mong đợi và đáp ứng các yêu cầu. Một mục tiêu khác của test này có thể là cung cấp thông tin cho các bên liên quan về rủi ro của phát hành hệ thống tại một thời điểm nhất định.

1.1.2 Testing and Debugging

Testing and debugging are different.

- Executing tests can show failures that are caused by defects in the software - Thực hiện các test có thể cho thấy các lỗi được gây ra bởi các defect trong phần mềm.
- Debugging is the development activity that finds, analyzes, and fixes such defects - Gỡ lỗi là hoạt động phát triển tìm, phân tích và sửa các lỗi đó
- Subsequent confirmation testing checks whether the fixes resolved the defects - Test xác nhận xem các bản sửa lỗi có giải quyết được chính xác lỗi không.

1.2 Why is Testing Necessary?

- Rigorous testing of systems and documentation can help to reduce the risk of problems occurring during operation - Test một cách cẩn trọng HT và tài liệu có thể giúp giảm rủi ro cho các vấn đề có thể xảy ra trong quá trình vận hành PM
- When defects are detected, and subsequently fixed, this contributes to the quality of the components or systems - Khi phát hiện lỗi và sau đó được sửa, điều này góp phần vào chất lượng của các thành phần hoặc hệ thống
- Software testing may also be required to meet contractual or legal requirements, or industry-specific standards - Test PM cũng là 1 yêu cầu trong hợp đồng và các yêu cầu pháp lý hoặc chuẩn công nghiệp.

1.2.1 Testing's Contributions to Success

- It is quite common for software and systems to be delivered into operation and, due to the presence of defects, to subsequently cause failures or otherwise not meet the stakeholders' needs - Các phần mềm và hệ thống được đưa vào vận hành bị gặp lỗi, sau đó gây ra failures hoặc không đáp ứng nhu cầu của các bên liên quan.
- Using appropriate test techniques can reduce the frequency of such problematic deliveries, when those techniques are applied with the appropriate level of test expertise, in the appropriate test levels,

and at the appropriate points in the software development lifecycle - Sử dụng các kỹ thuật test phù hợp có thể làm giảm tần suất của các lần bàn giao hàng có vấn đề, khi các kỹ thuật đó được áp dụng với mức độ test, giai đoạn test và tại các điểm thích hợp trong vòng đời phát triển phần mềm.

- Testing contributes to overall software development and maintenance success - Test góp phần phát triển phần mềm tổng thể và bảo trì thành công.

Example:

Phase	**Contributes**
Testers involved in requirements reviews or user story refinement could detect defects Tester tham gia vào đánh giá yêu cầu hoặc sàng lọc yêu cầu người dùng có thể phát hiện lỗi	Reduces the risk of incorrect or untestable functionality being developed. Giảm nguy cơ chức năng không chính xác hoặc không thể kiểm chứng đang được phát triển.
Testers work closely with system designers while the system is being designed can increase understanding and how to test it Tester làm việc chặt chẽ với các designer trong khi hệ thống đang được thiết kế có thể tăng sự hiểu biết và cách test nó	Reduce the risk of fundamental design defects and enable tests to be identified at an early stage. Giảm nguy cơ lỗi thiết kế cơ bản và cho phép các xét nghiệm được xác định ở giai đoạn đầu.
Testers work closely with developers while the code can increase understanding and how to test it Tester làm việc chặt chẽ với các developer trong khi code có thể tăng sự hiểu biết và cách test nó	Reduce the risk of defects within the code and the tests. Giảm nguy cơ lỗi trong code và các bài test.
Testers verify and validate the software prior to release can detect failures Tester xác minh và xác thực phần mềm trước khi phát hành có thể phát hiện lỗi	Increases the likelihood that the software meets stakeholder needs and satisfies requirements. Tăng khả năng phần mềm đáp ứng nhu cầu của các bên liên quan và đáp ứng các yêu cầu.

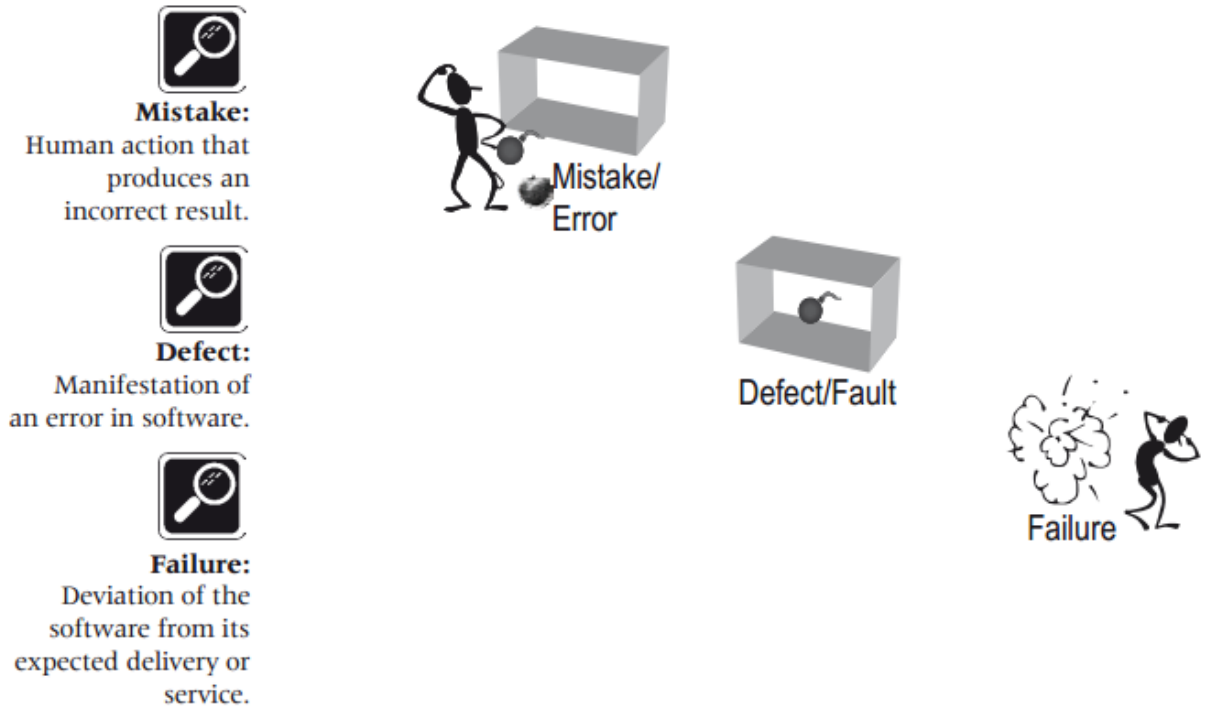
1.2.2 Quality Assurance and Testing

- Quality assurance and testing are not the same, but they are related - Đảm bảo chất lượng và test không giống nhau, nhưng chúng có liên quan
- Quality management includes all activities that direct and control an organization with regard to quality, includes both quality assurance and quality control - Quản lý chất lượng bao gồm tất cả các hoạt động chỉ đạo và kiểm soát một tổ chức liên quan đến chất lượng, bao gồm cả đảm bảo chất lượng và kiểm soát chất lượng
- Quality assurance is typically focused on adherence to proper processes, in order to provide confidence that the appropriate levels of quality will be achieved. - Đảm bảo chất lượng thường

tập trung vào việc tuân thủ các quy trình thích hợp, để đảm bảo rằng mức độ chất lượng phù hợp sẽ đạt được.

- Quality assurance contributes to defect prevention (example: the use of root cause analysis to detect and remove the causes of defects, retrospective meetings to improve processes) - Đảm bảo chất lượng góp phần ngăn ngừa defect (ví dụ: việc sử dụng phân tích nguyên nhân gốc để phát hiện và loại bỏ các nguyên nhân gây ra lỗi, các cuộc họp để cải thiện các quy trình)
- Quality control involves various activities, including test activities, that support the achievement of appropriate levels of quality. Test activities are part of the overall software development or maintenance process - Kiểm soát chất lượng liên quan đến các hoạt động khác nhau, bao gồm các hoạt động test, hỗ trợ đạt được mức chất lượng phù hợp. Các hoạt động test là một phần của quy trình bảo trì hoặc phát triển phần mềm tổng thể
- Since quality assurance is concerned with the proper execution of the entire process, quality assurance supports proper testing - Vì đảm bảo chất lượng liên quan đến việc thực hiện đúng toàn bộ quy trình, đảm bảo chất lượng hỗ trợ test thích hợp

1.2.3 Errors, Defects, and Failures



Errors may occur for many reasons, such as:

- Time pressure Human fallibility - Áp lực thời gian
- Inexperienced or insufficiently skilled project participants - Người tham gia dự án thiếu kinh nghiệm hoặc không đủ kỹ năng
- Miscommunication between project participants, including miscommunication about requirements and design - Thông tin sai lệch giữa những người tham gia dự án, bao gồm cả thông tin sai lệch về các yêu cầu và thiết kế

- Complexity of the code, design, architecture, the underlying problem to be solved, and/or the technologies used - Độ phức tạp của code, thiết kế, kiến trúc, vấn đề cơ bản cần giải quyết và / hoặc các công nghệ được sử dụng
- Misunderstandings about intra-system and inter-system interfaces, especially when such intra-system and inter-system interactions are large in number - Những hiểu lầm về giao diện giữa hệ thống và liên hệ thống, đặc biệt là khi các tương tác giữa hệ thống và liên hệ thống đó có số lượng lớn
- New, unfamiliar technologies - Công nghệ mới, lạ

In addition to failures caused due to defects in the code, failures can also be caused by environmental conditions - Ngoài các lỗi gây ra do lỗi trong code, các lỗi cũng có thể được gây ra bởi các điều kiện môi trường

Not all unexpected test results are failures- Không phải tất cả các kết quả test không mong muốn đều là failure:

- False positives may occur due to errors in the way tests were executed, or due to defects in the test data, the test environment, or other testware, or for other reasons. The inverse situation can also occur, where similar errors or defects lead to false negatives – lỗi giả có thể xảy ra do sai trong cách thực hiện các test hoặc do lỗi trong dữ liệu test, môi trường test hoặc các phần mềm test khác hoặc vì lý do khác. Tình huống nghịch đảo cũng có thể xảy ra, trong đó các lỗi hoặc defect tương tự dẫn đến lỗi sai.

1.2.4 Defects, Root Causes and Effects

The root causes of defects are the earliest actions or conditions that contributed to creating the defects - Nguyên nhân gốc rễ của defect là những hành động hoặc điều kiện sớm nhất góp phần tạo ra nhiều defect

Identify the root causes of failure can- Xác định nguyên nhân gốc rễ của sự failure có thể:

- Reduce the occurrence of similar defects in the future - Giảm sự xuất hiện của các defect tương tự trong tương lai.
- Lead to process improvements that prevent a significant number of future defects from being introduced - Dẫn đến cải tiến quy trình ngăn chặn một số lượng đáng kể các defect trong tương lai

1.3 Seven Principles of Testing – 7 nguyên lý cơ bản của testing

Testing shows presence of defects

- Testing can show that defects are present, but cannot prove that there are no defects - Test có thể chỉ ra lỗi, nhưng không thể chứng minh rằng phần mềm không có lỗi.
- Testing reduces the probability of undiscovered defects remaining in a software but, even if no defects are found, it is not a proof of correctness - Test làm giảm xác suất của các lỗi chưa được khám phá vẫn còn trong phần mềm nhưng ngay cả khi không có 1 lỗi nào được tìm thấy, nó cũng không phải là một bằng chứng về tính đúng đắn phần mềm.

Exhaustive testing is impossible – Complete test

- Test everything (all combination of inputs and preconditions) is not feasible - Test tất cả mọi thứ (tất cả các kết hợp của đầu vào và điều kiện) là không khả thi

- Instead of exhaustive testing, risk analysis and priorities should be used to focus testing efforts - Thay vì test tất cả, phân tích rủi ro và sắp xếp thứ tự ưu tiên nên được sử dụng để tập trung trong testing.

Early testing

- Testing activities should start as early as possible in the software or software development life cycle and should focus on defined objectives - Các hoạt động test nên bắt đầu càng sớm càng tốt trong chu kỳ phần mềm hoặc toàn bộ vòng đời phát triển và nên tập trung vào mục tiêu được xác định
- Perform the test design and review activities early can find defects early on when they are cheap to find and fix - Thực hiện các test và review càng sớm thì lỗi càng được phát hiện sớm khi đó ít tốn công để để tìm và sửa chữa.

Defect clustering: defect density

- A small numbers of modules usually contains most of the defects discovered during pre-release testing, or is responsible for most of the operational failures Một số lượng nhỏ của các mô-đun thường có chứa hầu hết các lỗi được phát hiện trong quá trình test trước khi phát hành, hoặc là chịu trách nhiệm cho hầu hết các failure của hệ thống.
- Rule 80/20: Module core often contains 80% defects - Quy tắc 80/20: Module lõi thường chứa 80% lỗi

Pesticide paradox

- If the same tests are repeated over and over again, no new defects can be found - Nếu các test tương tự được lặp đi lặp lại nhiều lần, không có lỗi mới nào có thể được tìm thấy.
- To overcome this pesticide paradox, test cases need to be regularly reviewed and revised; new and different tests need to be written to exercise different parts of the software to find potentially more defects - Để khắc phục nghịch lý thuốc trừ sâu này, các test case cần phải được thường xuyên rà soát và sửa đổi; Test mới và khác đi để tìm ra nhiều lỗi tiềm ẩn hơn.

Testing is context dependent

- Testing is done differently in different context - Test được thực hiện khác nhau trong bối cảnh khác nhau
- Ex: Safety-critical software is tested differently from an ecommerce site - Ví dụ: phần mềm an toàn quan trọng được test khác nhau với một trang web thương mại điện tử

Absence of error fallacy

- All specified requirements and fixing all defects found could still produce a system that is difficult to use, that does not fulfill the users' needs and expectations, or that is inferior compared to other competing systems - tất cả các yêu cầu được chỉ định và sửa tất cả các lỗi được tìm thấy vẫn có thể tạo ra một hệ thống khó sử dụng, không đáp ứng nhu cầu và mong đợi của người dùng, hoặc kém hơn so với các hệ thống cạnh tranh khác.

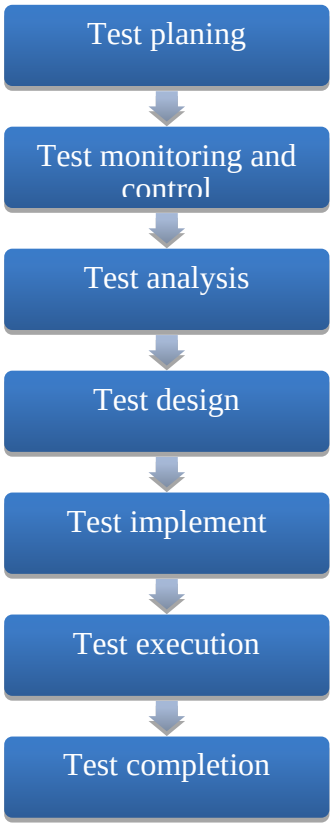
1.4 Test Process

1.4.1 Test Process in Context

Contextual factors that influence the test process for an organization, include, but are not limited to - Các yếu tố bối cảnh ảnh hưởng đến quá trình test cho một tổ chức, bao gồm, nhưng không giới hạn ở:

- Software development lifecycle model and project methodologies being used - Mô hình vòng đời phát triển phần mềm và phương pháp dự án đang được sử dụng
    - Test levels and test types being considered - Mức test và loại test đang được xem xét
    - Product and project risks - Rủi ro sản phẩm và dự án
    - Business domain - Lĩnh vực kinh doanh
- Operational constraints, including but not limited to - Các ràng buộc hoạt động, bao gồm nhưng không giới hạn ở:
    - Budgets and resources - Ngân sách và tài nguyên
    - Timescales - Thời gian
    - Complexity - Phức tạp
    - Contractual and regulatory requirements - Yêu cầu hợp đồng và quy định
- Organizational policies and practices - Chính sách và thực hành tổ chức
- Required internal and external standards - Yêu cầu tiêu chuẩn nội bộ và bên ngoài

1.4.2 Test Activities and Tasks



In Agile development involves small iterations of software design, build, and test that happen on a continuous basis, supported by on-going planning. So test activities are also happening on an iterative, continuous basis within this development approach. – Trong Agile bao gồm các giai đoạn lặp nhỏ của

thiết kế, xây dựng và test phần mềm xảy ra liên tục, được hỗ trợ bởi kế hoạch liên tục. Vì vậy, các hoạt động test cũng đang diễn ra trên cơ sở lặp đi lặp lại, liên tục trong phương pháp phát triển này.

In sequential development, the stepped logical sequence of activities will involve overlap, combination, concurrency, or omission, so tailoring these main activities within the context of the system and the project is usually required. - Trong phát triển tuần tự, các hoạt động thực hiện có thể là tuần tự, chồng chéo, kết hợp, đồng thời hoặc thiếu sót, do đó, điều chỉnh các hoạt động chính này trong bối cảnh của hệ thống và dự án thường được yêu cầu.

Test activities	Objectives	Work products
Test planning	- Define the objectives of testing and the approach for meeting test objectives (e.g., specifying suitable test techniques and tasks, and formulating a test schedule for meeting a deadline). - May be revisited based on feedback from monitoring and control activities	One or more test plans includes: - information about the test basis - to which the other test work products will be related via traceability information - exit criteria (or definition of done) which will be used during test monitoring and control
Test monitoring and control	- Involves the on-going comparison of actual progress against the test plan using any test monitoring metrics defined in the test plan - Involves taking actions necessary to meet the objectives of the test plan - Are supported by the evaluation of exit criteria for test execution as part of a given test level may include: + Checking test results and logs against specified coverage criteria + Assessing the level of component or system quality based on test results and logs + Determining if more tests are needed - Provide test progress reports, including deviations from the plan and information to support any decision to stop testing	- test progress reports (produced on an ongoing and/or a regular basis) - test summary reports (produced at various completion milestones)
Test analysis	Analyzing the test basis appropriate to the test level being considered: - Requirement specifications - Design and implementation information - The implementation of the component or system - Risk analysis reports Evaluating the test basis and test items to	- defined and prioritized test conditions - bi-directionally traceable - test charters - reporting of defects in the test basis

	identify defects of various types, such as: - Ambiguities - Omissions - Inconsistencies - Inaccuracies - Contradictions - Superfluous statements Identifying features and sets of features to be tested - Defining and prioritizing test conditions (functional, non-functional, and structural characteristics, other business and technical factors, and levels of risks) - Capturing bi-directional traceability between each element of the test basis and the associated test conditions	
Test design	- Designing and prioritizing test cases and sets of test cases - Identifying necessary test data to support test conditions and test cases - Designing the test environment and identifying any required infrastructure and tools - Capturing bi-directional traceability between the test basis, test conditions, test cases, and test procedures	- High-level test cases, without concrete values for input data and expected results - Bi-directionally traceable to the test condition(s) it covers. - Test data, the design of the test environment, and the identification of infrastructure and tools, though the extent to which these results are documented varies significantly. - Test conditions defined in test analysis may be further refined in test design
Test implementation	- Developing and prioritizing test procedures, and, potentially, creating automated test scripts - Creating test suites from the test procedures and (if any) automated test scripts - Arranging the test suites within a test execution schedule in a way that results in	- Test procedures and the sequencing of those test procedures - Test suites - A test execution schedule - The test data serve to assign concrete values to the inputs and expected results of test

	efficient test execution - Building the test environment and verifying that everything needed has been set up correctly - Preparing test data and ensuring it is properly loaded in the test environment - Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test suites	cases - The concrete expected results which are associated with concrete test data are identified by using a test oracle.
Test execution	- Recording the IDs and versions of the test item(s) or test object, test tool(s), and testware - Executing tests either manually or by using test execution tools - Comparing actual results with expected results - Analyzing anomalies to establish their likely causes - Reporting defects based on the failures observed - Logging the outcome of test execution (e.g., pass, fail, blocked) - Repeating test activities either as a result of action taken for an anomaly, or as part of the planned testing - Verifying and updating bi-directional traceability between the test basis, test conditions, test cases, test procedures, and test results.	- Documentation of the status of individual test cases or test procedures (e.g., ready to run, pass, fail, blocked, deliberately skipped, etc.) - Defect reports - Documentation about which test item(s), test object(s), test tools, and testware were involved in the testing
Test completion	- Checking whether all defect reports are closed, entering change requests or product backlog items for any defects that remain unresolved at the end of test execution	- Test summary reports - Action items for improvement of subsequent projects or iterations - Change requests or product

	- Creating a test summary report to be communicated to stakeholders - Finalizing and archiving the test environment, the test data, the test infrastructure, and other testware for later reuse - Handing over the testware to the maintenance teams, other project teams, and/or other stakeholders who could benefit from its use - Analyzing lessons learned from the completed test activities to determine changes needed for future iterations, releases, and projects - Using the information gathered to improve test process maturity	backlog items - Finalized testware.

1.4.4 Traceability between the Test Basis and Test Work Products

In order to implement effective test monitoring and control, it is important to establish and maintain traceability throughout the test process between each element of the test basis and the various test work products associated with that element - để thực hiện giám sát và kiểm soát test hiệu quả, điều quan trọng là phải thiết lập và duy trì khả năng truy nguyên trong suốt quá trình test giữa từng yếu tố của cơ sở test và các sản phẩm công việc test khác nhau được liên kết với yếu tố đó

Good traceability supports - Hỗ trợ truy xuất nguồn gốc tốt:

- Analyzing the impact of changes - Phân tích tác động của những thay đổi
- Making testing auditable - Làm cho test có thể test được
- Meeting IT governance criteria - Đáp ứng tiêu chí quản trị CNTT
- Improving the understandability of test progress reports and test summary reports to include the status of elements of the test basis (e.g., requirements that passed their tests, requirements that failed their tests, and requirements that have pending tests) - Cải thiện tính dễ hiểu của báo cáo tiến độ test và báo cáo tóm tắt test để bao gồm trạng thái của các yếu tố của cơ sở test (ví dụ: các yêu cầu đã vượt qua test của chúng, yêu cầu không vượt qua test và yêu cầu có các test đang chờ xử lý)
- Relating the technical aspects of testing to stakeholders in terms that they can understand - Liên quan các khía cạnh kỹ thuật của test với các bên liên quan theo các điều khoản mà họ có thể hiểu

- Providing information to assess product quality, process capability, and project progress against business goals - Cung cấp thông tin để đánh giá chất lượng sản phẩm, khả năng xử lý và tiến độ dự án so với mục tiêu kinh doanh

1.5 The Psychology of Testing

1.5.1 Human Psychology and Testing

Identifying defects may be perceived as criticism of the product and of its author. An element of human psychology called confirmation bias can make it difficult to accept information that disagrees with currently held beliefs - Xác định các defect có thể được coi là sự chỉ trích về sản phẩm và tác giả của nó. Một yếu tố của tâm lý con người được gọi là thiên vị có thể gây khó khăn cho việc chấp nhận thông tin không đồng ý với niềm tin đang có

Some people may perceive testing as a destructive activity, even though it contributes greatly to project progress and product quality - Một số người có thể coi test là một hoạt động phá hoại, mặc dù nó đóng góp rất lớn vào tiến độ dự án và chất lượng sản phẩm

To reduce these perceptions - Để giảm những nhận thức này:

- Information about defects and failures should be communicated in a constructive way - Thông tin về defect và failure nên được truyền đạt theo cách xây dựng
- Good interpersonal skills to be able to communicate effectively about defects, failures, test results, test progress, and risks, and to build positive relationships with colleagues - Kỹ năng giao tiếp tốt để có thể giao tiếp hiệu quả về khuyết điểm, failure, kết quả test, tiến độ test và rủi ro và xây dựng mối quan hệ tích cực với đồng nghiệp

- Good communicate:
    - Start with collaboration rather than battles. Remind everyone of the common goal of better quality systems - Bắt đầu với sự hợp tác chứ không phải trận chiến. Nhắc nhở mọi người về mục tiêu chung của các hệ thống chất lượng tốt hơn.
    - Emphasize the benefits of testing - Nhấn mạnh lợi ích của test
    - Communicate test results and other findings in a neutral, fact-focused way without criticizing the person who created the defective item - Truyền đạt kết quả test và các phát hiện khác theo cách trung lập, tập trung vào thực tế mà không chỉ trích người tạo ra sản phẩm bị lỗi.
    - Try to understand how the other person feels and the reasons they may react negatively to the information - Cố gắng hiểu cảm giác của người khác và lý do họ có thể phản ứng tiêu cực với thông tin.

- Confirm that the other person has understood what has been said and vice versa - Xác nhận rằng người kia đã hiểu những gì đã nói và ngược lại.

- Clearly defining the right set of test objectives has important psychological implications - Xác định rõ ràng đúng mục tiêu test có ý nghĩa tâm lý quan trọng

1.5.2 Tester's and Developer's Mindsets

Developers and testers often think differently

	Developer	Tester
Objective	design and build a product	verifying and validating the product, finding defects prior to release
Mindset	more interested in designing and building solutions than in contemplating what might be wrong with those solutions difficult to find mistakes in their own work developers should be able to test their own code quan tâm nhiều hơn đến việc thiết kế và xây dựng các giải pháp hơn là suy ngẫm những gì có thể sai với những giải pháp đó khó tìm thấy sai lầm trong công việc của họ developer có thể test code của riêng họ	curiosity, professional pessimism, a critical eye, attention to detail, and a motivation for good and positive communications and relationships tò mò, bi quan chuyên nghiệp, một con mắt quan trọng, chú ý đến chi tiết và một động lực cho các mối quan hệ và giao tiếp tốt và tích cực

Some of the test activities done by independent testers- Một số hoạt động test được thực hiện bởi tester độc lập:

- to increases defect detection effectiveness, which is particularly important for large, complex, or safety-critical systems - để tăng hiệu quả phát hiện defect, điều đặc biệt quan trọng đối với các hệ thống lớn, phức tạp hoặc yêu cầu độ an toàn cao.
- they have different cognitive biases from the authors - họ có những thành kiến nhận thức khác nhau từ các tác giả

Chapter 2: Testing Throughout The Software Development Lifecycle

2.1 Software Development Lifecycle Models

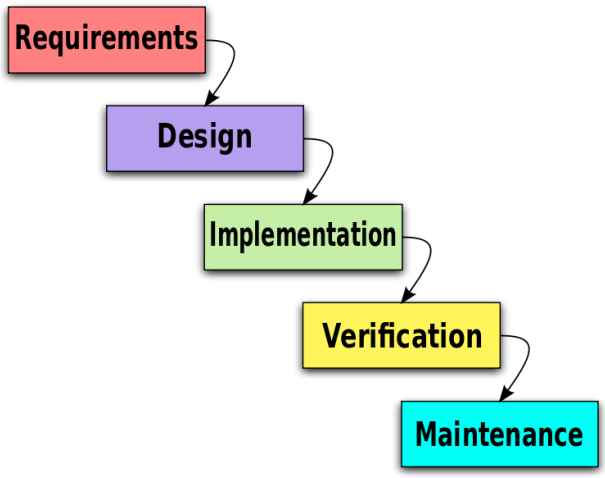
In any software development lifecycle model, there are several characteristics of good testing - Trong bất kỳ mô hình vòng đời phát triển phần mềm nào, có một số đặc điểm của test tốt:

- For every development activity, there is a corresponding test activity - Đối với mọi hoạt động phát triển, có một hoạt động test tương ứng
- Each test level has test objectives specific to that level - Mỗi test level có mục tiêu test cụ thể cho cấp độ đó
- Test analysis and design for a given test level begin during the corresponding development activity - Phân tích và thiết kế test cho một mức test nhất định bắt đầu trong hoạt động phát triển tương ứng
- Testers participate in discussions to define and refine requirements and design, and are involved in reviewing work products (e.g., requirements, design, user stories, etc.) as soon as drafts are available - Người test tham gia thảo luận để xác định và tinh chỉnh các yêu cầu và thiết kế, và tham gia vào việc xem xét các sản phẩm công việc (ví dụ: yêu cầu, thiết kế, yêu cầu của người dùng, v.v.) ngay khi có bản nháp

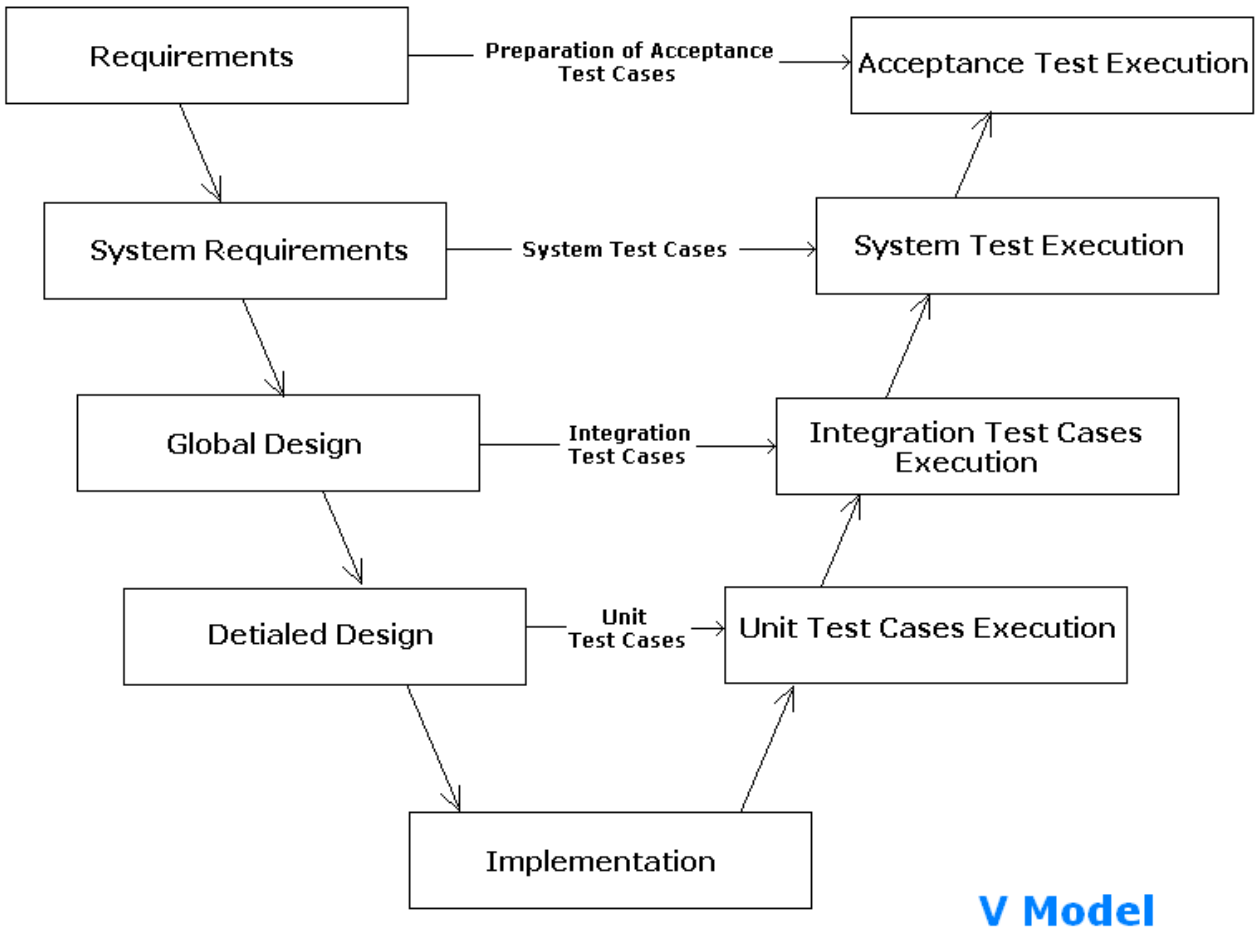
In any model, test activities should start in the early stages of the lifecycle, adhering to the testing principle of early testing. There are two common models - Trong bất kỳ mô hình nào, các hoạt động test nên bắt đầu trong giai đoạn đầu của vòng đời, tuân thủ nguyên tắc của test sớm. Có hai mô hình phổ biến:

- Sequential development models
- Iterative and incremental development models

1. **Sequential development models**



In the Waterfall model, the development activities (e.g., requirements analysis, design, coding, testing) are completed one after another. In this model, test activities only occur after all other development activities have been completed - Trong mô hình Waterfall, các hoạt động phát triển (ví dụ: phân tích yêu cầu, thiết kế, code, test) được hoàn thành lần lượt. Trong mô hình này, các hoạt động test chỉ xảy ra sau khi tất cả các hoạt động phát triển khác đã được hoàn thành.

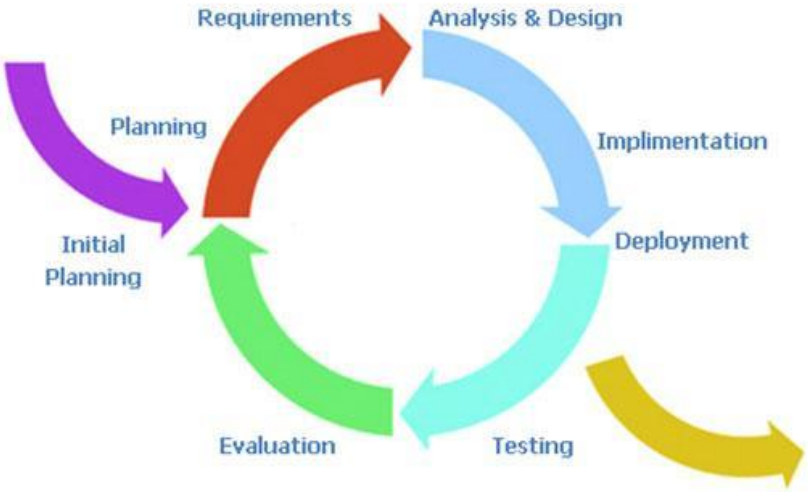


The V-model:

- integrates the test process throughout the development process - tích hợp quy trình test trong suốt quá trình phát triển
- early testing.
- includes test levels associated with each corresponding development phase - bao gồm các test level liên quan đến từng giai đoạn phát triển tương ứng
- some cases, test levels overlapping occurs - một số trường hợp, mức độ test chồng chéo xảy ra

Sequential development models deliver software that contains the complete set of features, but typically require months or years for delivery to stakeholders and users - Các mô hình phát triển tuần tự cung cấp phần mềm chứa toàn bộ tính năng hoàn chỉnh, nhưng thường yêu cầu nhiều tháng hoặc nhiều năm để bàn giao cho các bên liên quan và người dùng.

2. Iterative and incremental development models



Examples of Incremental development - Ví dụ:

- Rational Unified Process: long iterations (e.g., two to three months), and the feature increments are correspondingly large, such as two or three groups of related features - các lần lặp dài (ví dụ: hai đến ba tháng) và các mức tăng tính năng tương ứng lớn, chẳng hạn như hai hoặc ba nhóm tính năng liên quan
- Scrum: short iterations (e.g., hours, days, or a few weeks), and the feature increments are correspondingly small, such as a few enhancements and/or two or three new features - các lần lặp ngắn (ví dụ: giờ, ngày hoặc vài tuần) và các mức tăng tính năng tương ứng nhỏ, chẳng hạn như một vài cải tiến và / hoặc hai hoặc ba tính năng mới
- Kanban: Implemented with or without fixed-length iterations, which can deliver either a single enhancement or feature upon completion, or can group features together to release at once - Được triển khai có hoặc không có các lần lặp có độ dài cố định, có thể cung cấp một cải tiến hoặc tính năng duy nhất khi hoàn thành hoặc có thể nhóm các tính năng lại với nhau để phát hành cùng một lúc
- Spiral (or prototyping): Involves creating experimental increments, some of which may be heavily re-worked or even abandoned in subsequent development work - Liên quan đến việc tạo ra các gia số test, một số trong đó có thể được làm lại rất nhiều hoặc thậm chí bị bỏ rơi trong công việc phát triển tiếp theo

Incremental development has some characteristics - có một số đặc điểm:

- Establishing requirements, designing, building, and testing a system in pieces, which means that the software's features grow incrementally - Thiết lập các yêu cầu, thiết kế, xây dựng và test một hệ thống thành từng phần, điều đó có nghĩa là các tính năng của phần mềm tăng dần.
- Groups of features are specified, designed, built, and tested together in a series of cycles, often of a fixed duration - Các nhóm tính năng được chỉ định, thiết kế, xây dựng và test cùng nhau trong một chuỗi các chu kỳ, thường có thời lượng cố định
- Involve changes to features developed in earlier iterations, along with changes in project scope - Liên quan đến các thay đổi đối với các tính năng được phát triển trong các lần lặp trước đó, cùng với các thay đổi trong phạm vi dự án

- Each feature is tested at several test levels as it moves towards delivery - Mỗi tính năng được test ở một số test level khi nó chuyển sang bàn giao.
- In some cases, teams use continuous delivery or continuous deployment, both of which involve significant automation of multiple test levels as part of their delivery pipelines - Trong một số trường hợp, các nhóm sử dụng bàn giao liên tục hoặc triển khai liên tục, cả hai đều liên quan đến tự động hóa đáng kể của nhiều test level như một phần của chuỗi bàn giao của họ
- Release to end-users on a feature-by-feature basis, on an iteration- by-iteration basis, or in a more traditional major-release fashion - Phát hành cho người dùng cuối trên cơ sở từng tính năng lặp đi lặp lại hoặc theo cách phát hành chính truyền thống hơn.
- Regression testing is increasingly important as the system grows - Test hồi quy ngày càng quan trọng khi hệ thống phát triển.

Iterative and incremental models may deliver usable software in weeks or even days, but may only deliver the complete set of requirements product over a period of months or even years - Các mô hình lặp và gia tăng có thể cung cấp phần mềm có thể sử dụng được trong vài tuần hoặc thậm chí vài ngày, nhưng chỉ có thể cung cấp bộ sản phẩm yêu cầu hoàn chỉnh trong khoảng thời gian vài tháng hoặc thậm chí nhiều năm.

2.1.2 Software Development Lifecycle Models in Context

Software development lifecycle models must be selected and adapted to the context of project and product characteristics - Các mô hình vòng đời phát triển phần mềm phải được chọn và điều chỉnh phù hợp với bối cảnh của của dự án và sản phẩm:

- the project goal - mục tiêu dự án
- the type of product being developed - loại sản phẩm đang được phát triển
- business priorities (e.g., time-to-market) - các ưu tiên kinh doanh (ví dụ: thời gian tiếp thị)
- identified product and project risks - các rủi ro về dự án hoặc sản phẩm

Depending on the context of the project, it may be necessary to combine or reorganize test levels and/or test activities - Tùy thuộc vào bối cảnh của dự án, có thể cần phải kết hợp hoặc sắp xếp lại các test level và / hoặc các hoạt động test

2.2 Test Levels (K2)

The test levels:

- Component testing
- Integration testing
- System testing
- Acceptance testing

Test levels are characterized by the following attributes:

- Specific objectives
- Test basis, referenced to derive test cases Test object (i.e., what is being tested)
- Typical defects and failures
- Specific approaches and responsibilities
- A suitable test environment is required

Common objectives of test levels:

- Reducing risk
- Verifying whether the functional and non-functional behaviors
- Building confidence that changes have not broken existing components, systems
- Finding defects
- Preventing defects from escaping to higher test levels

Item	**Component testing**	**Integration Testing**	**System Testing**
Objective	focuses on components that are separately testable	focuses on interactions and interfaces between components or systems	focuses on the behavior and capabilities of a whole system or product
Special	Done in isolation from the rest of the system, require mock objects, service virtualization, harnesses, stubs, and drivers.	two different levels of integration testing: Component integration testing System integration testing	Incompleted or undocumented specification
Test types	Functionality (e.g., correctness of calculations) Non-functional characteristics (e.g., searching for memory leaks) Structural properties (e.g., decision testing).	Functional and non-functional	Functional and non-functional and data quality characteristic
Test basis	Detailed design Code Data model Component specifications Test objects	Software and system design Sequence diagrams Interface and communication protocol specifications Use cases Architecture at component or system level Workflows External interface definitions	System and software requirement specifications (functional and non-functional) Risk analysis reports Use cases Epics and user stories Models of system behavior State diagrams System and user manuals
Typical test objects	Components, units or modules Code and data structures Classes	Subsystems Databases Infrastructure Interfaces APIs	Applications Hardware/software systems

	Database modules	Microservices	Operating systems System under test (SUT) System configuration and configuration data
Environment	Development environment with framework, debug tool,...	Specific environment	correspond to the production environment
Typical defects and failures	Incorrect functionality (e.g., not as described in design specifications) Data flow problems Incorrect code and logic Defects are typically fixed as soon as they are found	Inconsistent message structures between systems Incorrect data, missing data, or incorrect data encoding Incorrect sequencing or timing of interface calls Interface mismatch Failures in communication between components Unhandled or improperly handled communication failures between components Incorrect assumptions about the meaning, units, or boundaries of the data being passed between component Failure to comply with mandatory security regulations	Incorrect calculations Incorrect or unexpected system functional or non-functional behavior Incorrect control and/or data flows within the system Failure to properly and completely carry out end-to-end functional tasks Failure of the system to work properly in the production environment(s) Failure of the system to work as described in system and user manuals
Responsibilities	Developer	Developer and tester	Independent testers
Approach	Test-first approach Test driven development (TDD)	The greater the scope of integration, the more difficult it becomes to isolate defects to a specific component or system **Big-bang integration** **Incremental integration**	the most appropriate techniques for the aspect(s) of the system to be tested

2.2.4 Acceptance Testing

Objectives:

- Establishing confidence in the quality of the system as a whole - Thiết lập niềm tin vào chất lượng của toàn bộ hệ thống
- Validating that the system is complete and will work as expected - Xác nhận rằng hệ thống đã hoàn tất và sẽ hoạt động như mong đợi
- Verifying that functional and non-functional behaviors of the system are as specified - Xác minh rằng các hành vi chức năng và phi chức năng của hệ thống được quy định

Common forms of acceptance testing:

1. **User acceptance testing (UAT)**

The acceptance testing of the system by users is typically focused on validating the fitness for use of the system by intended users in a real or simulated operational environment. The main objective is building confidence that the users can use the system to meet their needs, fulfill requirements, and perform business processes with minimum difficulty, cost, and risk - Việc test chấp nhận hệ thống của người dùng thường tập trung vào việc xác thực tính phù hợp để sử dụng hệ thống, bởi người dùng test trong môi trường hoạt động thực tế hoặc mô phỏng. Mục tiêu chính là xây dựng niềm tin rằng người dùng có thể sử dụng hệ thống để đáp ứng nhu cầu của họ, đáp ứng các yêu cầu và thực hiện các quy trình kinh doanh với độ khó, chi phí và rủi ro tối thiểu.

2. **Operational acceptance testing (OAT)**

The acceptance testing of the system by operations or systems administration staff is usually performed in a (simulated) production environment. The tests focus on operational aspects, and may include - Việc test chấp nhận hệ thống bởi các nhân viên quản trị hệ thống hoặc vận hành thường được thực hiện trong môi trường sản xuất (mô phỏng). Các test tập trung vào các khía cạnh hoạt động và có thể bao gồm:

- Testing of backup and restore
- Installing, uninstalling and upgrading
- Disaster recovery
- User management
- Maintenance tasks
- Data load and migration tasks
- Checks for security vulnerabilities
- Performance testing

The main objective of operational acceptance testing is building confidence that the operators or system administrators can keep the system working properly for the users in the operational environment, even under exceptional or difficult conditions - Mục tiêu chính của test chấp nhận vận hành là xây dựng niềm

tin rằng các nhà khai thác hoặc quản trị viên hệ thống có thể giữ cho hệ thống hoạt động tốt cho người dùng trong môi trường hoạt động, ngay cả trong các điều kiện đặc biệt hoặc khó khăn

3. Contractual and regulatory acceptance testing

Contractual acceptance testing is performed against a contract's acceptance criteria for producing custom-developed software. Acceptance criteria should be defined when the parties agree to the contract - Test chấp nhận theo hợp đồng được thực hiện theo tiêu chí chấp nhận của hợp đồng để sản xuất phần mềm được phát triển tùy chỉnh. Tiêu chí chấp nhận nên được xác định khi các bên đồng ý với hợp đồng.

Regulatory acceptance testing is performed against any regulations that must be adhered to, such as government, legal, or safety regulations - Test chấp nhận theo quy định được thực hiện đối với bất kỳ quy định nào phải tuân thủ, chẳng hạn như các quy định của chính phủ, pháp lý hoặc an toàn

Contractual acceptance testing and regulatory acceptance testing are often performed by users or by independent testers - Test chấp nhận hợp đồng và test chấp nhận theo quy định thường được thực hiện bởi người dùng hoặc bởi người test độc lập

4. Alpha and beta testing

Alpha and beta testing are typically used by developers of commercial off-the-shelf (COTS) software who want to get feedback from potential or existing users, customers, and/or operators before the software product is put on the market - Test alpha và beta thường được sử dụng bởi các nhà phát triển phần mềm thương mại (COTS), những người muốn nhận phản hồi từ người dùng tiềm năng hoặc khách hàng hiện tại, khách hàng và / hoặc nhà khai thác trước khi sản phẩm phần mềm được đưa ra thị trường.

Alpha testing is performed at the developing organization's site, not by the development team, but by potential or existing customers, and/or operators or an independent test team - Test Alpha được thực hiện tại môi trường của tổ chức phát triển, không phải bởi nhóm phát triển, mà bởi các khách hàng tiềm năng hoặc khách hàng hiện tại.

Beta testing is performed by potential or existing customers, and/or operators at their own locations. Beta testing may come after alpha testing, or may occur without any preceding alpha testing having occurred - Test Beta được thực hiện bởi các khách hàng tiềm năng hoặc hiện có tại các địa điểm của họ. Test beta có thể đến sau khi test alpha, hoặc có thể xảy ra mà không có bất kỳ test alpha nào trước đó đã xảy ra

Test basis

- Business processes
- User or business requirements
- Regulations, legal contracts and standards
- Use cases
- System requirements
- System or user documentation
- Installation procedures
- Risk analysis reports

In addition, as a test basis for deriving test cases for operational acceptance testing, one or more of the following work products can be used:

- Backup and restore procedures
- Disaster recovery procedures
- Non-functional requirements
- Operations documentation
- Deployment and installation instructions
- Performance targets
- Database packages
- Security standards or regulations

Typical test objects

- System under test
- System configuration and configuration data
- Business processes for a fully integrated system
- Recovery systems and hot sites (for business continuity and disaster recovery testing)
- Operational and maintenance processes
- Forms Reports
- Existing and converted production data

Typical defects and failures

- System workflows do not meet business or user requirements
- Business rules are not implemented correctly
- System does not satisfy contractual or regulatory requirements
- Non-functional failures such as security vulnerabilities, inadequate performance efficiency under high loads, or improper operation on a supported platform

Specific approaches and responsibilities

Acceptance testing may also occur at other times, for example:

In a sequential development lifecycle

- Acceptance testing of a COTS software product may occur when it is installed or integrated - Test chấp nhận sản phẩm phần mềm COTS có thể xảy ra khi được cài đặt hoặc tích hợp
- Acceptance testing of a new functional enhancement may occur before system testing - Test chấp nhận tăng cường chức năng mới có thể xảy ra trước khi test hệ thống

In iterative development

- Acceptance testing during and at the end of each iteration, it focused on verifying a new feature against its acceptance criteria and those focused on validating that a new feature satisfies the users' needs - Kiểm tra chấp nhận trong và cuối mỗi lần lặp, nó tập trung vào xác minh một tính

năng mới theo các tiêu chí chấp nhận của nó và những người tập trung vào xác thực rằng một tính năng mới đáp ứng nhu cầu của người dùng.

- Alpha tests and beta tests may occur, either at the end of each iteration, after the completion of each iteration, or after a series of iterations - Các test Alpha và test beta có thể xảy ra, vào cuối mỗi lần lặp, sau khi hoàn thành mỗi lần lặp hoặc sau một loạt các lần lặp
- User acceptance tests, operational acceptance tests, regulatory acceptance tests, and contractual acceptance tests also may occur, either at the close of each iteration, after the completion of each iteration, or after a series of iterations - Các test chấp nhận của người dùng, test chấp nhận vận hành, test chấp nhận theo quy định và test chấp nhận hợp đồng cũng có thể xảy ra, vào cuối mỗi lần lặp, sau khi hoàn thành mỗi lần lặp hoặc sau một loạt các lần lặp

2.3 Test Types

- Evaluating functional quality characteristics, such as completeness, correctness, and appropriateness - Đánh giá các đặc tính chất lượng chức năng, chẳng hạn như tính đầy đủ, tính chính xác và tính phù hợp
- Evaluating non-functional quality characteristics, such as reliability, performance efficiency, security, compatibility, and usability - Đánh giá các đặc tính chất lượng phi chức năng, chẳng hạn như độ tin cậy, hiệu quả hoạt động, bảo mật, khả năng tương thích và khả năng sử dụng
- Evaluating whether the structure or architecture of the component or system is correct, complete, and as specified - Đánh giá xem cấu trúc hoặc kiến trúc của thành phần hoặc hệ thống là chính xác, đầy đủ và theo quy định
- Evaluating the effects of changes, such as confirming that defects have been fixed (confirmation testing) and looking for unintended changes in behavior resulting from software or environment changes (regression testing) - Đánh giá tác động của các thay đổi, chẳng hạn như xác nhận rằng các lỗi đã được sửa (kiểm tra xác nhận) và tìm kiếm các thay đổi ngoài ý muốn trong hành vi do thay đổi phần mềm hoặc môi trường (kiểm tra hồi quy)

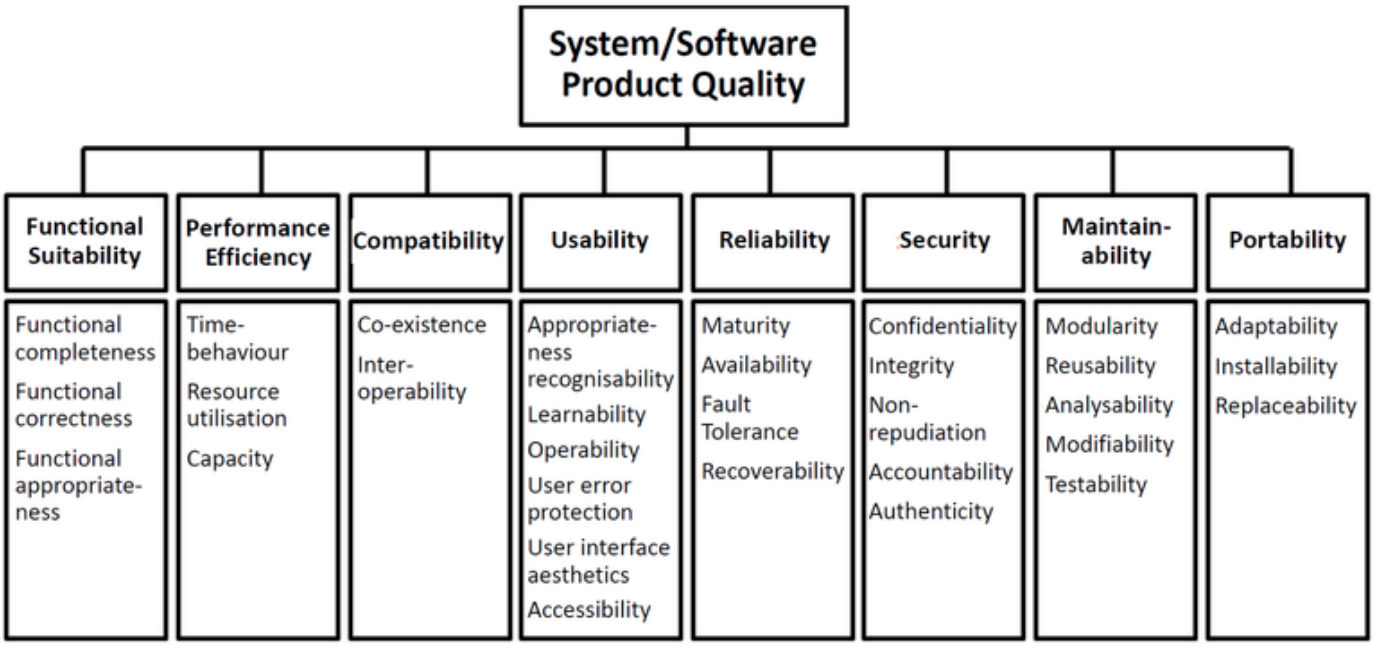
2.3.1 Functional Testing

Functional testing of a system involves tests that evaluate functions that the system should perform. Functional requirements may be described in work products such as business requirements specifications, epics, user stories, use cases, or functional specifications, or they may be undocumented. The functions are “what” the system should do - Kiểm tra chức năng của một hệ thống bao gồm các kiểm tra đánh giá các chức năng mà hệ thống nên thực hiện. Các yêu cầu chức năng có thể được mô tả trong các sản phẩm công việc như thông số kỹ thuật yêu cầu kinh doanh, sử thi, câu chuyện của người dùng, trường hợp sử dụng hoặc thông số chức năng hoặc chúng có thể không có giấy tờ. Các chức năng là những gì mà hệ thống nên làm

2.3.2 Non-functional Testing

Non-functional testing of a system evaluates characteristics of systems and software such as usability, performance efficiency or security - Kiểm tra phi chức năng của một hệ thống đánh giá các đặc điểm của hệ thống và phần mềm như khả năng sử dụng, hiệu quả hoạt động hoặc bảo mật

ISO standard (ISO/IEC 25010)



2.3.3 White-box Testing

White-box testing derives tests based on the system's internal structure or implementation - Test hộp trắng xuất phát các test dựa trên cấu trúc hoặc triển khai bên trong của hệ thống

Structural coverage is the extent to which some type of structural element has been exercised by tests, and is expressed as a percentage of the type of element being covered - Độ bao phủ cấu trúc là mức độ mà một số loại yếu tố cấu trúc đã được thực hiện bằng các test và được biểu thị bằng phần trăm của loại yếu tố được bảo hiểm

2.3.4 Change-related Testing

- Confirmation testing: confirm whether the original defect has been successfully fixed - Kiểm tra xác nhận: xác nhận xem lỗi ban đầu đã được sửa thành công chưa.
- Regression testing: a change made in one part of the code, may accidentally affect the behavior of other parts of the code. Such unintended side-effects are called regressions. Regression testing involves running tests to detect such unintended side-effects. Changes may include changes to the environment, such as a new version of an operating system or database management system - Kiểm tra hồi quy: một thay đổi được thực hiện trong một phần của mã, có thể vô tình ảnh hưởng đến hành vi của các phần khác của mã. Tác dụng phụ ngoài ý muốn như vậy được gọi là hồi quy. Kiểm tra hồi quy bao gồm chạy test để phát hiện các tác dụng phụ ngoài ý muốn. Các thay đổi có thể bao gồm các thay đổi đối với môi trường, chẳng hạn như phiên bản mới của hệ điều hành hoặc hệ thống quản lý cơ sở dữ liệu.

2.4 Maintenance Testing

Maintenance testing focuses on testing the changes to the system, as well as testing unchanged parts that might have been affected by the changes. Maintenance can involve planned releases and unplanned releases (hot fixes) -Kiểm tra bảo trì tập trung vào kiểm tra các thay đổi đối với hệ thống, cũng như kiểm tra các bộ phận không thay đổi có thể bị ảnh hưởng bởi các thay đổi. Bảo trì có thể liên quan đến các bản phát hành theo kế hoạch và bản phát hành không có kế hoạch (bản sửa lỗi nóng).

A maintenance release may require maintenance testing at multiple test levels, using various test types, based on its scope. The scope of maintenance testing depends on - Một bản phát hành bảo trì có thể yêu cầu test bảo trì ở nhiều test level, sử dụng các loại test khác nhau, dựa trên phạm vi của nó. Phạm vi kiểm tra bảo trì phụ thuộc vào:

- The degree of risk of the change, for example, the degree to which the changed area of software communicates with other components or systems
- The size of the existing system
- The size of the change

2.4.1 Triggers for Maintenance

We can classify the triggers for maintenance as follows - Chúng tôi có thể phân loại các kích hoạt để bảo trì như sau:

- Modification, such as planned enhancements (e.g., release-based), corrective and emergency changes, changes of the operational environment (such as planned operating system or database upgrades), upgrades of COTS software, and patches for defects and vulnerabilities - Sửa đổi, chẳng hạn như cải tiến theo kế hoạch (ví dụ: dựa trên bản phát hành), thay đổi khắc phục và khẩn cấp, thay đổi môi trường vận hành (như nâng cấp hệ điều hành hoặc nâng cấp cơ sở dữ liệu), nâng cấp phần mềm COTS và bản vá lỗi và lỗ hổng
- Migration, such as from one platform to another, which can require operational tests of the new environment as well as of the changed software, or tests of data conversion when data from another application will be migrated into the system being maintained - Di chuyển, chẳng hạn như từ nền tảng này sang nền tảng khác, có thể yêu cầu kiểm tra hoạt động của môi trường mới cũng như phần mềm đã thay đổi hoặc test chuyển đổi dữ liệu khi dữ liệu từ ứng dụng khác sẽ được di chuyển vào hệ thống được duy trì
- Retirement, such as when an application reaches the end of its life - Nghỉ hưu, chẳng hạn như khi một ứng dụng đến hết tuổi thọ của nó

2.4.2 Impact Analysis for Maintenance

Impact analysis evaluates the changes that were made for a maintenance release to identify the intended consequences as well as expected and possible side effects of a change, and to identify the areas in the system that will be affected by the change - Phân tích tác động đánh giá các thay đổi được thực hiện cho một bản phát hành bảo trì để xác định các hậu quả dự kiến cũng như các tác dụng phụ có thể xảy ra và có thể xảy ra của một thay đổi và để xác định các khu vực trong hệ thống sẽ bị ảnh hưởng bởi thay đổi.

Impact analysis can also help to identify the impact of a change on existing tests. The side effects and affected areas in the system need to be tested for regressions, possibly after updating any existing tests affected by the change - Phân tích tác động cũng có thể giúp xác định tác động của một thay đổi đối với các test hiện tại. Các tác dụng phụ và các khu vực bị ảnh hưởng trong hệ thống cần phải được kiểm tra hồi quy, có thể sau khi cập nhật bất kỳ test hiện có nào bị ảnh hưởng bởi thay đổi.

Impact analysis may be done before a change is made, to help decide if the change should be made, based on the potential consequences in other areas of the system - Phân tích tác động có thể được thực hiện trước khi thay đổi được thực hiện, để giúp quyết định xem có nên thực hiện thay đổi hay không, dựa trên các hậu quả tiềm ẩn trong các lĩnh vực khác của hệ thống.

Impact analysis can be difficult if - Phân tích tác động có thể khó khăn nếu:

- Specifications (e.g., business requirements, user stories, architecture) are out of date or missing Test cases are not documented or are out of date - Thông số kỹ thuật (ví dụ: yêu cầu nghiệp vụ, câu chuyện của người dùng, kiến trúc) đã hết hạn hoặc thiếu Các trường hợp kiểm tra không được ghi lại hoặc đã hết hạn
- Bi-directional traceability between tests and the test basis has not been maintained - Truy nguyên nguồn gốc hai chiều giữa các xét nghiệm và cơ sở test chưa được duy trì
- Tool support is weak or non-existent - Công cụ hỗ trợ yếu hoặc không tồn tại
- The people involved do not have domain and/or system knowledge - Những người liên quan không có kiến thức về miền và / hoặc hệ thống
- Insufficient attention has been paid to the software's maintainability during development - Sự quan tâm không đầy đủ đã được trả cho khả năng bảo trì của phần mềm trong quá trình phát triển

Chapter 3 Static Testing

3.1 Static Testing Basics

Static testing relies on the manual examination of work products (i.e., reviews) or tool-driven evaluation of the code or other work products (i.e., static analysis). Both types of static testing assess the code or other work product being tested without actually executing the code or work product being tested - Test tĩnh dựa trên việc kiểm tra thủ công các sản phẩm công việc (tức là, đánh giá) hoặc đánh giá dựa trên công cụ của mã hoặc các sản phẩm công việc khác (tức là, phân tích tĩnh). Cả hai loại test tĩnh đều đánh giá mã hoặc sản phẩm công việc khác đang được test mà không thực sự thực thi mã hoặc sản phẩm công việc đang được test

3.1.1 Work Products that Can Be Examined by Static Testing

Almost any work product can be examined using static testing (reviews and/or static analysis) - Hầu hết mọi sản phẩm công việc đều có thể được kiểm tra bằng test tĩnh (đánh giá và / hoặc phân tích tĩnh)

Reviews can be applied to any work product that the participants know how to read and understand - Đánh giá có thể được áp dụng cho bất kỳ sản phẩm công việc nào mà người tham gia biết cách đọc và hiểu.

Static analysis can be applied efficiently to any work product with a formal structure (typically code or models) for which an appropriate static analysis tool exists - Phân tích tĩnh có thể được áp dụng hiệu quả cho bất kỳ sản phẩm công việc nào có cấu trúc chính thức (thường là mã hoặc mô hình) có công cụ phân tích tĩnh thích hợp.

Static analysis can even be applied with tools that evaluate work products written in natural language such as requirements (e.g., checking for spelling, grammar, and readability) - Phân tích tĩnh thậm chí có thể được áp dụng với các công cụ đánh giá các sản phẩm công việc được viết bằng ngôn ngữ tự nhiên, chẳng hạn như các yêu cầu (ví dụ: kiểm tra chính tả, ngữ pháp và khả năng đọc).

3.1.2 Benefits of Static Testing

- Detecting and correcting defects more efficiently, and prior to dynamic test execution - Phát hiện và sửa lỗi hiệu quả hơn và trước khi thực hiện kiểm tra động
- Identifying defects which are not easily found by dynamic testing - Xác định các lỗi không dễ dàng tìm thấy bằng test động
- Preventing defects in design or coding by uncovering inconsistencies, ambiguities, contradictions, omissions, inaccuracies, and redundancies in requirements - Ngăn ngừa các khiếm khuyết trong thiết kế hoặc code bằng cách phát hiện ra sự không nhất quán, mơ hồ, mâu thuẫn, thiếu sót, không chính xác và dự phòng trong các yêu cầu
- Increasing development productivity (e.g., due to improved design, more maintainable code) - Tăng năng suất phát triển (ví dụ: do thiết kế được cải tiến, mã dễ bảo trì hơn)
- Reducing development cost and time - Giảm chi phí và thời gian phát triển

- Reducing testing cost and time - Giảm chi phí và thời gian test
- Reducing total cost of quality over the software's lifetime, due to fewer failures later in the lifecycle or after delivery into operation - Giảm tổng chi phí chất lượng trong suốt vòng đời phần mềm, do ít lỗi hơn trong vòng đời hoặc sau khi vận hành vào hoạt động
- Improving communication between team members in the course of participating in reviews - Cải thiện giao tiếp giữa các thành viên trong nhóm trong quá trình tham gia đánh giá

3.1.3 Differences between Static and Dynamic Testing

Compare with dynamic testing, static testing can - So sánh với test động, test tĩnh có thể:

- provide an assessment of the quality of the work products - cung cấp một đánh giá về chất lượng của các sản phẩm làm việc
- identify defects as early as possible - xác định khuyết điểm càng sớm càng tốt
- finds defects in work products directly rather than identifying failures caused by defects when the software is run - tìm thấy các lỗi trong các sản phẩm công việc trực tiếp thay vì xác định các lỗi gây ra bởi các lỗi khi phần mềm được chạy.
- find the defect with much less effort - tìm ra khuyết điểm với nỗ lực ít hơn nhiều
- used to improve the consistency and internal quality of work products - được sử dụng để cải thiện tính nhất quán và chất lượng nội bộ của sản phẩm làm việc

Typical defects that are easier and cheaper to find and fix through static testing include - Các lỗi điển hình dễ tìm và rẻ hơn để tìm và sửa chữa thông qua kiểm tra tĩnh bao gồm:

- Requirement defects (e.g., inconsistencies, ambiguities, contradictions, omissions, inaccuracies, and redundancies) - Lỗi yêu cầu (ví dụ: sự không nhất quán, sự mơ hồ, mâu thuẫn, thiếu sót, không chính xác và dư thừa)
- Design defects (e.g., inefficient algorithms or database structures, high coupling, low cohesion) - Lỗi thiết kế (ví dụ: thuật toán không hiệu quả hoặc cấu trúc cơ sở dữ liệu, khớp nối cao, độ gắn kết thấp)
- Coding defects (e.g., variables with undefined values, variables that are declared but never used, unreachable code, duplicate code) - Lỗi code (ví dụ: các biến có giá trị không xác định, các biến được khai báo nhưng không bao giờ được sử dụng, mã không thể truy cập, mã trùng lặp)
- Deviations from standards (e.g., lack of adherence to coding standards) - Độ lệch so với tiêu chuẩn (ví dụ: thiếu tuân thủ các tiêu chuẩn code)
- Incorrect interface specifications (e.g., different units of measurement used by the calling system than by the called system) - Thông số kỹ thuật giao diện không chính xác (ví dụ: các đơn vị đo lường khác nhau được sử dụng bởi hệ thống gọi so với hệ thống được gọi)
- Security vulnerabilities (e.g., susceptibility to buffer overflows) - Các lỗ hổng bảo mật (ví dụ: dễ bị tràn bộ đệm)

- Gaps or inaccuracies in test basis traceability or coverage (e.g., missing tests for an acceptance criterion) - Các lỗ hổng hoặc không chính xác trong truy xuất nguồn gốc hoặc phạm vi bảo hiểm (ví dụ: các test còn thiếu cho một tiêu chí chấp nhận)
- Maintainability defects (e.g., improper modularization, poor reusability of components, code that is difficult to analyze and modify without introducing new defects) - Lỗi bảo trì (ví dụ: mô đun hóa không phù hợp, khả năng tái sử dụng kém của các thành phần, mã khó phân tích và sửa đổi mà không đưa ra lỗi mới)

3.2 Review Process

3.2.1 Work Product Review Process

Planning

- Defining the scope, which includes the purpose of the review, what documents or parts of documents to review, and the quality characteristics to be evaluated - Xác định phạm vi, bao gồm mục đích của đánh giá, tài liệu hoặc bộ phận nào của tài liệu cần xem xét và các đặc điểm chất lượng cần đánh giá
- Estimating effort and timeframe - Ước tính nỗ lực và khung thời gian
- Identifying review characteristics such as the review type with roles, activities, and checklists - Xác định các đặc điểm đánh giá, chẳng hạn như loại đánh giá với vai trò, hoạt động và danh sách kiểm tra
- Selecting the people to participate in the review and allocating roles - Chọn người tham gia đánh giá và phân bổ vai trò
- Defining the entry and exit criteria for more formal review types (e.g., inspections) - Xác định tiêu chí vào và ra cho các loại đánh giá chính thức hơn (ví dụ: kiểm tra)
- Checking that entry criteria are met (for more formal review types) - Kiểm tra các tiêu chí đầu vào được đáp ứng (đối với các loại đánh giá chính thức hơn)

Initiate review

- Distributing the work product (physically or by electronic means) and other material, such as issue log forms, checklists, and related work products - Bàn giao sản phẩm công việc (vật lý hoặc bằng phương tiện điện tử) và các tài liệu khác, chẳng hạn như biểu mẫu nhật ký, danh sách kiểm tra và các sản phẩm công việc liên quan
- Explaining the scope, objectives, process, roles, and work products to the participants - Giải thích phạm vi, mục tiêu, quy trình, vai trò và sản phẩm công việc cho người tham gia
- Answering any questions that participants may have about the review - Trả lời bất kỳ câu hỏi mà người tham gia có thể có về đánh giá
  Đánh giá cá nhân (tức là, chuẩn bị cá nhân)

Individual review (i.e., individual preparation)

- Reviewing all or part of the work product - Xem xét tất cả hoặc một phần của sản phẩm làm việc
- Noting potential defects, recommendations, and questions - Lưu ý các khiếm khuyết tiềm năng, khuyến nghị và câu hỏi

Issue communication and analysis

- Communicating identified potential defects (e.g., in a review meeting) - Truyền đạt các khiếm khuyết tiềm ẩn đã được xác định (ví dụ: trong một cuộc họp đánh giá)
- Analyzing potential defects, assigning ownership and status to them - Phân tích các khiếm khuyết tiềm ẩn, giao quyền sở hữu và trạng thái cho chúng
- Evaluating and documenting quality characteristics - Đánh giá và ghi nhận các đặc tính chất lượng
- Evaluating the review findings against the exit criteria to make a review decision (reject; major changes needed; accept, possibly with minor changes) - Đánh giá kết quả đánh giá theo các tiêu chí xuất cảnh để đưa ra quyết định đánh giá (từ chối; cần thay đổi lớn; chấp nhận, có thể với những thay đổi nhỏ)

Fixing and reporting

- Creating defect reports for those findings that require changes - Tạo báo cáo lỗi cho những phát hiện cần thay đổi
- Fixing defects found (typically done by the author) in the work product reviewed - Sửa các lỗi được tìm thấy (thường được thực hiện bởi tác giả) trong sản phẩm công việc được xem xét
- Communicating defects to the appropriate person or team (when found in a work product related to the work product reviewed) - Truyền đạt các khiếm khuyết cho người hoặc nhóm thích hợp (khi được tìm thấy trong một sản phẩm công việc liên quan đến sản phẩm công việc được xem xét)
- Recording updated status of defects (in formal reviews), potentially including the agreement of the comment originator - Ghi lại trạng thái cập nhật của lỗi (trong các đánh giá chính thức), có thể bao gồm cả sự đồng ý của người khởi tạo bình luận
- Gathering metrics (for more formal review types) - Thu thập số liệu (đối với các loại đánh giá chính thức hơn)
- Checking that exit criteria are met (for more formal review types) - Kiểm tra các tiêu chí thoát được đáp ứng (đối với các loại đánh giá chính thức hơn)
- Accepting the work product when the exit criteria are reached - Chấp nhận sản phẩm làm việc khi đạt tiêu chí kết thúc

3.2.2 Roles and responsibilities in a formal review

Author

- Creates the work product under review
- Fixes defects in the work product under review (if necessary)

Management

- Is responsible for review planning
- Decides on the execution of reviews

- Assigns staff, budget, and time
- Monitors ongoing cost-effectiveness
- Executes control decisions in the event of inadequate outcomes

Facilitator (often called moderator)

- Ensures effective running of review meetings (when held) Mediates, if necessary, between the various points of view
- Is often the person upon whom the success of the review depends

Review leader

- Takes overall responsibility for the review
- Decides who will be involved and organizes when and where it will take place

Reviewers

- May be subject matter experts, persons working on the project, stakeholders with an interest in the work product, and/or individuals with specific technical or business backgrounds
- Identify potential defects in the work product under review
- May represent different perspectives (e.g., tester, programmer, user, operator, business analyst, usability expert, etc.)

Scribe (or recorder)

- Collates potential defects found during the individual review activity
- Records new potential defects, open points, and decisions from the review meeting (when held)

3.2.3 Review Types

Informal review (e.g., buddy check, pairing, pair review)

- Main purpose: detecting potential defects
- Possible additional purposes: generating new ideas or solutions, quickly solving minor problems
- Not based on a formal (documented) process
- May not involve a review meeting
- May be performed by a colleague of the author (buddy check) or by more people
- Results may be documented
- Varies in usefulness depending on the reviewers
- Use of checklists is optional
- Very commonly used in Agile development

Walkthrough

- Main purposes: find defects, improve the software product, consider alternative implementations, evaluate conformance to standards and specifications

- Possible additional purposes: exchanging ideas about techniques or style variations, training of participants, achieving consensus
- Individual preparation before the review meeting is optional
- Review meeting is typically led by the author of the work product
- Scribe is mandatory
- Use of checklists is optional
- May take the form of scenarios, dry runs, or simulations
- Potential defect logs and review reports may be produced
- May vary in practice from quite informal to very formal

Technical review

- Main purposes: gaining consensus, detecting potential defects
- Possible further purposes: evaluating quality and building confidence in the work product, generating new ideas, motivating and enabling authors to improve future work products, considering alternative implementations
- Reviewers should be technical peers of the author, and technical experts in the same or other disciplines
- Individual preparation before the review meeting is required
- Review meeting is optional, ideally led by a trained facilitator (typically not the author)
- Scribe is mandatory, ideally not the author
- Use of checklists is optional
- Potential defect logs and review reports are typically produced

Inspection

- Main purposes: detecting potential defects, evaluating quality and building confidence in the work product, preventing future similar defects through author learning and root cause analysis
- Possible further purposes: motivating and enabling authors to improve future work products and the software development process, achieving consensus
- Follows a defined process with formal documented outputs, based on rules and checklists
- Uses clearly defined roles
- Individual preparation before the review meeting is required
- Reviewers are either peers of the author or experts in other disciplines that are relevant to the work product
- Specified entry and exit criteria are used
- Scribe is mandatory
- Review meeting is led by a trained facilitator (not the author)
- Author cannot act as the review leader, reader, or scribe
-
- Potential defect logs and review report are produced
- Metrics are collected and used to improve the entire software development process, including the inspection process

3.2.4 Applying Review Techniques

Ad hoc

Reviewers are provided with little or no guidance on how this task should be performed - Người phản biện được cung cấp ít hoặc không có hướng dẫn về cách thực hiện nhiệm vụ này

Reviewers often read the work product sequentially, identifying and documenting issues as they encounter them - Người đánh giá thường đọc sản phẩm công việc một cách tuần tự, xác định và ghi lại các vấn đề khi họ gặp phải chúng.

Ad hoc reviewing is a commonly used technique needing little preparation - Xem xét ad hoc là một kỹ thuật thường được sử dụng cần ít sự chuẩn bị.

This technique is highly dependent on reviewer skills and may lead to many duplicate issues being reported by different reviewers - Kỹ thuật này phụ thuộc nhiều vào kỹ năng của người đánh giá và có thể dẫn đến nhiều vấn đề trùng lặp được báo cáo bởi những người đánh giá khác nhau.

Checklist-based

A checklist-based review is a systematic technique, whereby the reviewers detect issues based on checklists that are distributed at review initiation (e.g., by the facilitator) - Đánh giá dựa trên danh sách kiểm tra là một kỹ thuật có hệ thống, theo đó người đánh giá phát hiện các vấn đề dựa trên danh sách kiểm tra được bàn giao khi bắt đầu đánh giá (ví dụ: bởi người hướng dẫn).

A review checklist consists of a set of questions based on potential defects, which may be derived from experience - Danh sách kiểm tra đánh giá bao gồm một bộ câu hỏi dựa trên các khiếm khuyết tiềm ẩn, có thể xuất phát từ kinh nghiệm.

Checklists should be specific to the type of work product under review and should be maintained regularly to cover issue types missed in previous reviews - Danh sách kiểm tra phải cụ thể đối với loại sản phẩm công việc đang được xem xét và nên được duy trì thường xuyên để bao gồm các loại vấn đề bị bỏ lỡ trong các đánh giá trước đó

The main advantage of the checklist-based technique is a systematic coverage of typical defect types. Care should be taken not to simply follow the checklist in individual reviewing, but also to look for defects outside the checklist - Ưu điểm chính của kỹ thuật dựa trên danh sách kiểm tra là phạm vi bao phủ có hệ thống các loại khiếm khuyết điển hình. Cần thận trọng không chỉ đơn giản là làm theo danh sách kiểm tra trong đánh giá cá nhân, mà còn để tìm kiếm các khiếm khuyết bên ngoài danh sách kiểm tra.

Scenarios and dry runs

In a scenario-based review, reviewers are provided with structured guidelines on how to read through the work product - Trong đánh giá dựa trên kịch bản, người đánh giá được cung cấp các hướng dẫn có cấu trúc về cách đọc qua sản phẩm công việc.

A scenario-based approach supports reviewers in performing “dry runs” on the work product based on expected usage of the work product (if the work product is documented in a suitable format such as use cases) - Cách tiếp cận dựa trên kịch bản hỗ trợ các nhà đánh giá trong việc thực hiện các chương trình

chạy khô khô trên sản phẩm công việc dựa trên việc sử dụng sản phẩm công việc dự kiến (nếu sản phẩm công việc được ghi lại theo định dạng phù hợp như trường hợp sử dụng).

These scenarios provide reviewers with better guidelines on how to identify specific defect types than simple checklist entries - Các kịch bản này cung cấp cho người đánh giá các hướng dẫn tốt hơn về cách xác định các loại lỗi cụ thể hơn các mục nhập danh sách kiểm tra đơn giản.

As with checklist-based reviews, in order not to miss other defect types (e.g., missing features), reviewers should not be constrained to the documented scenarios - Như với các đánh giá dựa trên danh sách kiểm tra, để không bỏ lỡ các loại lỗi khác (ví dụ: các tính năng bị thiếu), người đánh giá không nên bị hạn chế đối với các tình huống được ghi lại.

Role-based

A role-based review is a technique in which the reviewers evaluate the work product from the perspective of individual stakeholder roles - Đánh giá dựa trên vai trò là một kỹ thuật trong đó người đánh giá đánh giá sản phẩm công việc từ góc độ vai trò của các bên liên quan

Typical roles include specific end user types (experienced, inexperienced, senior, child, etc.), and specific roles in the organization (user administrator, system administrator, performance tester, etc.) - Các vai trò điển hình bao gồm các loại người dùng cuối cụ thể (có kinh nghiệm, thiếu kinh nghiệm, cao cấp, trẻ em, v.v.) và các vai trò cụ thể trong tổ chức (quản trị viên người dùng, quản trị viên hệ thống, người kiểm tra hiệu suất, v.v.).

Perspective-based

In perspective-based reading, similar to a role-based review, reviewers take on different stakeholder viewpoints in individual reviewing - Trong cách đọc dựa trên quan điểm, tương tự như đánh giá dựa trên vai trò, người đánh giá đảm nhận các quan điểm của các bên liên quan khác nhau trong đánh giá cá nhân.

Typical stakeholder viewpoints include end user, marketing, designer, tester, or operations - Quan điểm của các bên liên quan điển hình bao gồm người dùng cuối, tiếp thị, nhà thiết kế, người test hoặc hoạt động.

Using different stakeholder viewpoints leads to more depth in individual reviewing with less duplication of issues across reviewers - Sử dụng các quan điểm của các bên liên quan khác nhau dẫn đến việc xem xét cá nhân sâu sắc hơn với ít vấn đề trùng lặp giữa các nhà đánh giá.

In addition, perspective-based reading also requires the reviewers to attempt to use the work product under review to generate the product they would derive from it. For example, a tester would attempt to generate draft acceptance tests if performing a perspective-based reading on a requirements specification to see if all the necessary information was included - Ngoài ra, việc đọc dựa trên quan điểm cũng yêu cầu người đánh giá cố gắng sử dụng sản phẩm công việc đang được xem xét để tạo ra sản phẩm mà họ sẽ lấy từ nó. Ví dụ, một người kiểm tra sẽ cố gắng tạo các test chấp nhận dự thảo nếu thực hiện đọc dựa trên phối cảnh trên một đặc tả yêu cầu để xem có bao gồm tất cả các thông tin cần thiết hay không.

Further, in perspective-based reading, checklists are expected to be used - Hơn nữa, trong việc đọc dựa trên quan điểm, danh sách kiểm tra dự kiến sẽ được sử dụng.

3.2.5 Success Factors for Reviews

- Each review has clear objectives, defined during review planning, and used as measurable exit criteria - Mỗi đánh giá có mục tiêu rõ ràng, được xác định trong quá trình lập kế hoạch đánh giá và được sử dụng làm tiêu chí thoát có thể đo lường được
- Review types are applied which are suitable to achieve the objectives and are appropriate to the type and level of software work products and participants - Các loại đánh giá được áp dụng phù hợp để đạt được các mục tiêu và phù hợp với loại và mức độ của các sản phẩm và người tham gia công việc phần mềm
- Any review techniques used, such as checklist-based or role-based reviewing, are suitable for effective defect identification in the work product to be reviewed - Bất kỳ kỹ thuật đánh giá nào được sử dụng, như đánh giá dựa trên danh sách kiểm tra hoặc dựa trên vai trò, đều phù hợp để xác định lỗi hiệu quả trong sản phẩm làm việc cần được xem xét
- Any checklists used address the main risks and are up to date - Bất kỳ danh sách kiểm tra được sử dụng giải quyết các rủi ro chính và được cập nhật
- Large documents are written and reviewed in small chunks, so that quality control is exercised by providing authors early and frequent feedback on defects - Tài liệu lớn được viết và xem xét trong các phần nhỏ, để kiểm soát chất lượng được thực hiện bằng cách cung cấp cho tác giả phản hồi sớm và thường xuyên về lỗi
- Participants have adequate time to prepare - Người tham gia có đủ thời gian chuẩn bị
- Reviews are scheduled with adequate notice - Đánh giá được lên lịch với thông báo đầy đủ
- Management supports the review process (e.g., by incorporating adequate time for review activities in project schedules) - Quản lý hỗ trợ quá trình xem xét (ví dụ: bằng cách kết hợp đủ thời gian cho các hoạt động đánh giá trong lịch trình dự án)

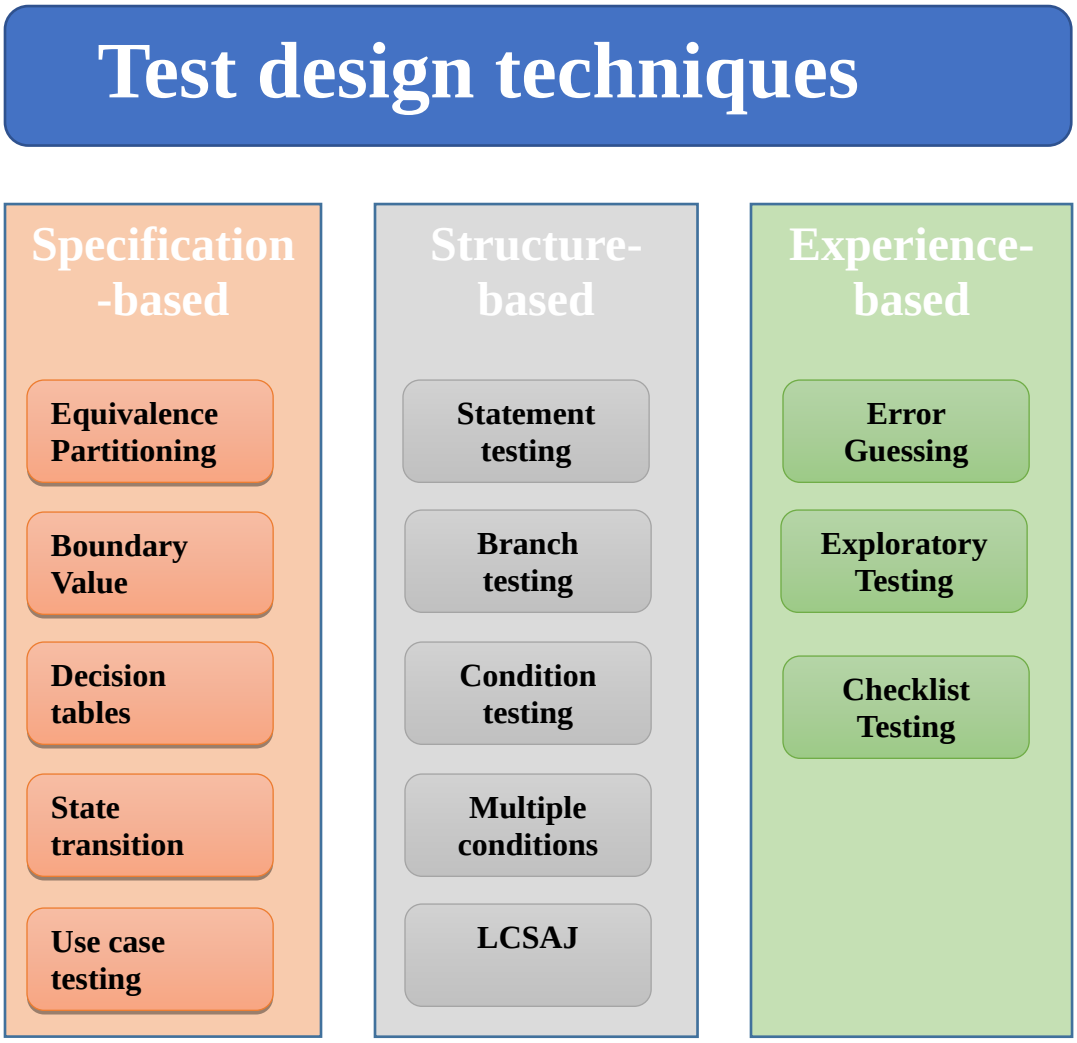
People-related success factors for reviews include - Các yếu tố thành công liên quan đến con người để đánh giá bao gồm:

- The right people are involved to meet the review objectives, for example, people with different skill sets or perspectives, who may use the document as a work input - Đúng người có liên quan để đáp ứng các mục tiêu đánh giá, ví dụ, những người có bộ kỹ năng hoặc quan điểm khác nhau, những người có thể sử dụng tài liệu làm đầu vào công việc
- Testers are seen as valued reviewers who contribute to the review and learn about the work product, which enables them to prepare more effective tests, and to prepare those tests earlier - Người kiểm tra được xem là người đánh giá có giá trị đóng góp cho việc đánh giá và tìm hiểu về sản phẩm công việc, cho phép họ chuẩn bị các bài kiểm tra hiệu quả hơn và chuẩn bị các bài kiểm tra đó sớm hơn
- Participants dedicate adequate time and attention to detail - Người tham gia dành thời gian thích hợp và chú ý đến chi tiết
- Reviews are conducted on small chunks, so that reviewers do not lose concentration during individual review and/or the review meeting (when held) - Đánh giá được tiến hành trên các phần nhỏ, để người đánh giá không bị mất tập trung trong quá trình đánh giá cá nhân và / hoặc cuộc họp đánh giá (khi được tổ chức)
- Defects found are acknowledged, appreciated, and handled objectively - Khiếm khuyết được tìm thấy được thừa nhận, đánh giá cao và xử lý khách quan

- The meeting is well-managed, so that participants consider it a valuable use of their time - Cuộc họp được quản lý tốt, để những người tham gia coi đó là một cách sử dụng có giá trị thời gian của họ
- The review is conducted in an atmosphere of trust; the outcome will not be used for the evaluation of the participants - Việc xem xét được tiến hành trong bầu không khí tin cậy; kết quả sẽ không được sử dụng để đánh giá những người tham gia
- Participants avoid body language and behaviors that might indicate boredom, exasperation, or hostility to other participants - Người tham gia tránh ngôn ngữ cơ thể và hành vi có thể biểu thị sự nhàm chán, bực tức hoặc thù địch với những người tham gia khác
- Adequate training is provided, especially for more formal review types such as inspections A culture of learning and process improvement is promoted - Đào tạo đầy đủ được cung cấp, đặc biệt đối với các loại đánh giá chính thức hơn như kiểm tra Văn hóa học tập và cải tiến quy trình được thúc đẩy

CHAPTER 4: Test Design Techniques

4.1 Categories of Test Techniques



4.1.1 Choosing Test Techniques

Depends on a number of factors:

- Type of component or system
- Component or system complexity
- Regulatory standards
- Customer or contractual requirements
- Risk levels
- Risk types
- Test objectives
- Available documentation
- Tester knowledge and skills

- Available tools
- Time and budget
- Software development lifecycle model
- Expected use of the software
- Previous experience with using the test techniques on the component or system to be tested
- The types of defects expected in the component or system

4.1.2 Categories of Test Techniques and Their Characteristics

Common characteristics of **specification-based test design** techniques include - Đặc điểm chung của kỹ thuật thiết kế test hướng đặc tả:

- Test conditions, test cases, and test data are derived from a test basis that may include software requirements, specifications, use cases, and user stories - Điều kiện test, trường hợp test và dữ liệu test được lấy từ cơ sở test có thể bao gồm các yêu cầu phần mềm, thông số kỹ thuật, trường hợp sử dụng và câu chuyện của người dùng
- Test cases may be used to detect gaps between the requirements and the implementation of the requirements, as well as deviations from the requirements - Các trường hợp test có thể được sử dụng để phát hiện các lỗ hổng giữa các yêu cầu và việc thực hiện các yêu cầu, cũng như các sai lệch so với các yêu cầu
- Coverage is measured based on the items tested in the test basis and the technique applied to the test basis - Độ che phủ được đo dựa trên các hạng mục được kiểm tra trong cơ sở test và kỹ thuật áp dụng cho cơ sở test

Common characteristics of **structure-based test design** techniques include - Đặc điểm chung của kỹ thuật thiết kế test hướng cấu trúc:

- Test conditions, test cases, and test data are derived from a test basis that may include code, software architecture, detailed design, or any other source of information regarding the structure of the software - Điều kiện test, trường hợp test và dữ liệu test được lấy từ cơ sở test có thể bao gồm mã, kiến trúc phần mềm, thiết kế chi tiết hoặc bất kỳ nguồn thông tin nào khác liên quan đến cấu trúc của phần mềm
- Coverage is measured based on the items tested within a selected structure (e.g., the code or interfaces) - Phạm vi được đo dựa trên các mục được kiểm tra trong một cấu trúc đã chọn (ví dụ: mã hoặc giao diện)
- Specifications are often used as an additional source of information to determine the expected outcome of test cases - Thông số kỹ thuật thường được sử dụng như một nguồn thông tin bổ sung để xác định kết quả dự kiến của các trường hợp test

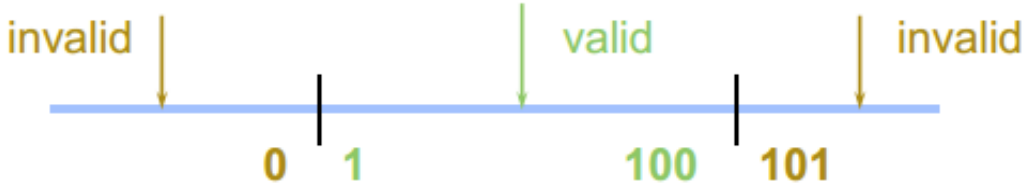
Common characteristics of **experience-based test design** techniques include - Đặc điểm chung của kỹ thuật thiết kế test hướng kinh nghiệm:

- Test conditions, test cases, and test data are derived from a test basis that may include knowledge and experience of testers, developers, users and other stakeholders - Điều kiện test, trường hợp test và dữ liệu test được lấy từ cơ sở test có thể bao gồm kiến thức và kinh nghiệm của người test, nhà phát triển, người dùng và các bên liên quan khác
- These techniques can be helpful in identifying tests that were not easily identified by other more systematic techniques. Depending on the tester's approach and experience, these techniques may achieve widely varying degrees of coverage and effectiveness - Những kỹ thuật này có thể hữu ích trong việc xác định các xét nghiệm không dễ xác định bằng các kỹ thuật có hệ thống khác. Tùy thuộc vào cách tiếp cận và kinh nghiệm của người test, các kỹ thuật này có thể đạt được mức độ bao phủ và hiệu quả khác nhau.
- Coverage can be difficult to assess and may not be measurable with these techniques - Độ che phủ có thể khó đánh giá và có thể không đo lường được bằng các kỹ thuật này

4.2 Black-box Test Techniques

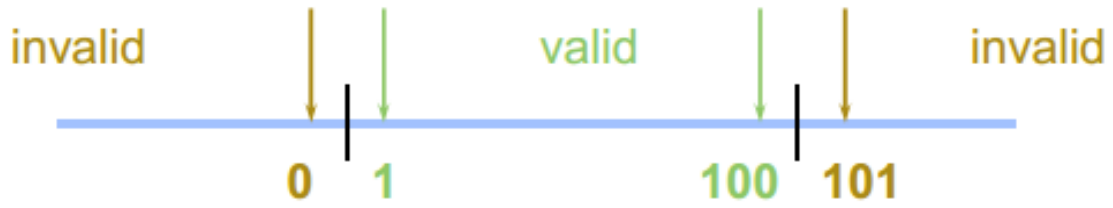
4.2.1 Equivalence partitioning (EP) - Phân vùng tương đương (EP)

- Divide (partition) the inputs, outputs, etc. into areas which are expected to exhibit similar behavior (equivalent) so they are likely to be processed in the same way. - Chia các vùng dựa vào đầu vào, đầu ra, vv thành các khu vực mà kết quả mong đợi là giống nhau (tương đương) vì thế chúng được xử lý theo 1 cách giống nhau
- Assumption: if one value works, all will work - Giả định: nếu một trong những giá trị làm đúng, tất cả các giá trị bên trong vùng đó sẽ làm đúng
- One from each partition better than all from one - Lấy một TH từ mỗi phân vùng tốt hơn so với lấy tất cả các TH
- Tests can be designed to cover all valid and invalid partitions. Equivalence partitioning is applicable at all levels of testing. - Test case được thiết kế để bao phủ tất cả các TH valid và invalid. EP được áp dụng ở tất cả các level test



4.2.2 Boundary value analysis (BVA)- Phân tích giá trị biên (BVA)

- Behavior at the edge of each equivalence partition is more likely to be incorrect than behavior within the partition, so boundaries are an area where testing is likely to yield defects - Lỗi có xu hướng ẩn nấp gần ranh giới
- Two-point boundary: The maximum and minimum values of a partition are its boundary values - Giá trị lớn nhất và nhỏ nhất trong 1 phân vùng là giá trị biên
- Three boundary values: the values before, at, and just over the boundary
- Tests can be designed to cover both valid and invalid boundary values. - Test được thiết kế để cover các giá trih biên valid và invalid
- Boundary value analysis can be applied at all test levels. It is relatively easy to apply and its defect- finding capability is high. - Phân tích giá trị biên được áp dụng ở tất cả các level test. Nó khá dễ dàng áp dụng để tìm ra lỗi.
- Detailed specifications are helpful in determining the interesting boundaries. - Đặc tả chi tiết sẽ giúp cho xác định được các giá trị biên này



4.2.3 Decision tables - bảng quyết định

- Explore combinations of inputs, situations or events

Khám phá sự kết hợp của các yếu tố đầu vào, tình huống hoặc sự kiện

- It is very easy to overlook specific combinations of input

Rất dễ dàng để bỏ qua sự kết hợp cụ thể của đầu vào

- Start by expressing the input conditions of interest so that they are TRUE or FALSE

Bắt đầu bằng cách diễn đạt các điều kiện đầu vào để họ có TRUE hoặc FALSE

Example: Student access – Phân quyền sinh viên

An university computer system allows students an allocation of disc space depending on their projects. - Một hệ thống máy tính của trường đại học cho phép sinh viên phân bổ không gian đĩa phụ thuộc vào các dự án của họ.

If they have used all their allotted space, they are only allowed restricted access, i.e. to delete files, not to create them. This is assuming they have logged on with a valid username and password. - Nếu họ đã sử

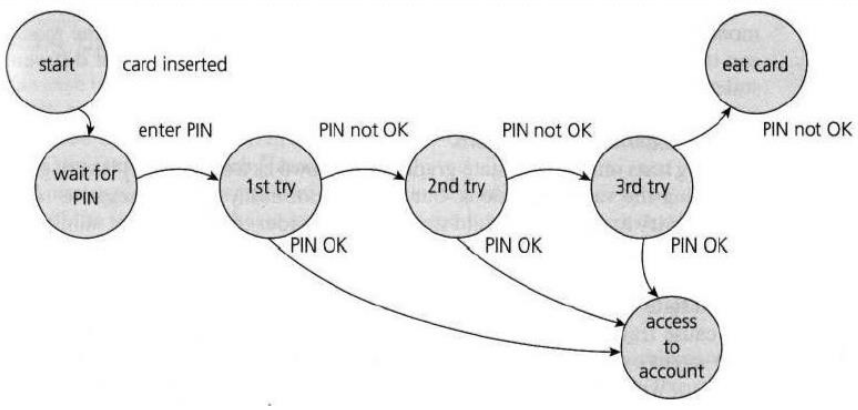
dụng tất cả các không gian được phân bổ của họ, họ chỉ được phép truy cập hạn chế, tức là để xóa các tập tin, không được tạo ra chúng. Đây là giả định họ đã đăng nhập với tên người dùng và mật khẩu hợp lệ.

Extending decision tables – Mở rộng của bảng ra quyết định

- Entries can be more than just ‘true’ or ‘false’ - Giá trị nhập vào có thể có nhiều hơn chỉ là 'đúng' hay 'sai'
  - Completing table needs to be done carefully - Hoàn thành bảng cần phải được thực hiện một cách cẩn thận
  - Rationalizing becomes more important - Hợp lý hoá trở nên quan trọng hơn

4.2.4 State transition testing - Test chuyển đổi trạng thái

- A system may exhibit a different response depending on current conditions or previous history (its state)
- Một hệ thống có thể biểu hiện một phản ứng khác nhau tùy thuộc vào điều kiện hiện tại hoặc tiền sử
- It allow the tester to view the software in terms of its states, transitions between states, the inputs or events that trigger state changes (transitions) and the states and inputs, and can highlight possible transitions that are invalid.
- Nó cho phép test để xem các phần mềm về trạng thái của nó, chuyển đổi giữa các trạng thái, các yếu tố đầu vào hoặc sự việc gây ra những thay đổi trạng thái (chuyển tiếp) và các lệnh và các đầu vào, và có thể làm nổi bật quá trình chuyển đổi hoặc có thể là không hợp lệ.
- Example: State diagram for PIN entry - Ví dụ: sơ đồ nhà nước cho nhập PIN



Switch:

1- Switch: single transition
2- Switch: couple transitions
3- Switch: triple transitions

A state transition model has four basic parts - Một mô hình chuyển trạng thái có bốn phần cơ bản::

- The states that the software may occupy (open/ closed or funded/ insufficient funds) - Các trạng thái mà phần mềm có thể thực hiện (mở / đóng cửa hoặc rút tiền / không đủ tiền)
- The transitions from one state to another (not all transitions are allowed) - Việc chuyển đổi từ một trạng thái khác (không phải tất cả các quá trình chuyển đổi được cho phép)
- The events that cause a transition (closing a file or withdrawing money) - Các sự kiện đã gây ra một quá trình chuyển đổi (đóng một tập tin hoặc rút tiền)
- The actions that result from a transition (an error message or being giving your cash) - Các hành động đó là kết quả của một quá trình chuyển đổi (một thông báo lỗi hoặc trả tiền cho bạn)

Note: One event can cause only one action, but that the same event – from a different state – may cause a different action and a different end state. - Lưu ý: 3Một sự kiện có thể gây ra chỉ có một hành động, nhưng là cùng một sự kiện - từ một trạng thái khác nhau - có thể gây ra một hành động khác nhau và một trạng thái kết thúc khác nhau.

- Test can be designed to cover a typical sequence of states, to cover every state, to exercise every transitions, to exercise specific sequences of transitions or to test invalid transitions - Test có thể được thiết kế để bao gồm một trình tự đặc trưng của các trạng thái, bao quát mọi giao dịch, thực hiện mỗi chuyển tiếp, để thực hiện trình tự cụ thể của quá trình chuyển đổi hoặc để test quá trình chuyển đổi không hợp lệ
- When designing test case, we should start with a typical scenario (a normal situation) then increase the coverage (every states/ every transitions) - Khi thiết kế trường hợp test, chúng ta nên bắt đầu với một kịch bản điển hình (một tình huống bình thường) sau đó tăng vùng phủ sóng (mỗi trạng thái / mỗi chuyển tiếp)
- One of the advantages of the state transition technique is that the model can be as detailed or as abstract as you need it to be. - Một trong những ưu điểm của kỹ thuật chuyển đổi trạng thái là mô hình có thể được chi tiết hoặc là trừu tượng như bạn cần nó làm.
- State transition testing is much used within the embedded software industry and technical automation in general - Test chuyển đổi trạng thái được sử dụng nhiều trong các ngành công nghiệp phần mềm nhúng và tự động hóa kỹ thuật nói chung.
- However, the technique is also suitable for modeling a business object having specific states or testing screen-dialog flows - Tuy nhiên, kỹ thuật này cũng là phù hợp với mô hình một đối tượng kinh doanh có các trạng thái cụ thể hoặc test các dòng màn hình-thoại

States table – Bảng trạng thái

In order to see the total number of combinations of states and transitions, both valid and invalid, a state table can be used. - Để xem tổng số kết hợp của trạng thái và quá trình chuyển đổi, hợp lệ và không hợp lệ, một bảng trạng thái có thể được sử dụng.

The table lists all the states down one side of the table and all the events that cause transitions along the top. Each cell then represents a state-event pair. The content of each cell indicates which state the system will move to. - Bảng liệt kê tất cả các trạng thái xuống một bên của bảng và tất cả các sự kiện gây ra quá trình chuyển đổi ở đầu trang. Mỗi ô sau đó đại diện cho một cặp trạng thái-sự kiện. Các nội dung của mỗi ô chỉ ra trạng thái hệ thống sẽ chuyển sang

	Insert card	Valid PIN	Invalid PIN
S1) Start state	S2	–	–
S2) Wait for PIN	–	S6	S3
S3) 1st try invalid	–	S6	S4
S4) 2nd try invalid	–	S6	S5
S5) 3rd try invalid	–	–	S7
S6) Access account	–	?	?
S7) Eat card	S1 (for new card)	–	–

4.2.5 Use case testing- Trường hợp sử dụng

- Use case testing is a technique that helps us identify test cases that exercise the whole system on a transaction by transaction basis from start to finish

  Sử dụng các trường hợp test là một kỹ thuật giúp chúng ta xác định các trường hợp test quyền thực hiện toàn bộ hệ thống trên cơ sở của giao dịch từ đầu đến cuối
- Use cases are a sequence of steps that describe the interactions between the actor (system or user) and the system

  Sử dụng các trường hợp là một chuỗi các bước mô tả sự tương tác giữa các đối tượng sử dụng (hệ thống hay người dùng) và hệ thống
- Use cases may be described at the abstract level (business use case, technology-free, business process level) or at the system level (system use case on the system functionality level)

  Sử dụng các trường hợp có thể được mô tả ở mức độ trừu tượng (trường hợp sử dụng kinh doanh, công nghệ miễn phí, mức độ quá trình kinh doanh) hoặc ở cấp độ hệ thống (trường hợp sử dụng ở cấp độ hệ thống chức năng)
- Each use case has preconditions which need to be met for a use case to work successfully

  Mỗi trường hợp sử dụng có điều kiện tiên quyết mà cần phải được đáp ứng cho một trường hợp sử dụng để làm việc thành công

- Each use case terminates with post conditions which are the observable results and final state of the system after the use case has been completed.
  Mỗi trường hợp sử dụng chấm dứt với điều kiện trước đó là những kết quả quan sát được và trạng thái cuối cùng của hệ thống sau khi các trường hợp sử dụng đã được hoàn thành.
- Each use case usually has a mainstream (or most likely) scenario and alternative paths.
  Mỗi trường hợp sử dụng thường có (hoặc có thể) kịch bản chính và đường dẫn thay thế.
- Use cases describe the “process flows” through a system based on its actual likely use, so the test case derived from use cases are most useful in uncovering defects in the process flows during real-world use of the system
  Sử dụng các trường hợp mô tả quá trình "chảy" qua một hệ thống dựa trên khả năng sử dụng thực tế của nó, vì vậy các trường hợp test xuất phát từ trường hợp sử dụng hữu ích nhất trong việc phát hiện các defect trong quá trình chạy qua khi sử dụng thực tế của hệ thống
- Use case testing also helps uncover integration defects caused by interaction and interference of different components, which individual component testing would not see
  Sử dụng các trường hợp test cũng giúp các defect hội nhập việc khám phá ra có vỏ bằng cách tương tác và giao thoa của các thành phần khác nhau, trong đó test thành phần cá nhân sẽ không nhìn thấy
- Designing test cases from use cases may be combined with other specification-based techniques
  Thiết kế trường hợp test từ trường hợp sử dụng có thể được kết hợp với các kỹ thuật dựa trên đặc điểm khác.

Example:

The below example explains the way to build a test case from Cash Withdrawal use case. Cash Withdraw is a function of ATM (Auto Teller Machine)

Basic Flow	*This Use Case begins with the ATM in the Ready State.* *Initiate Withdraw - Customer inserts bank card in the card reader on the ATM machine* *Verify Bank Card - The ATM reads the account code from the magnetic strip on the bank card and checks if it is an acceptable bank card.* *Enter PIN - The ATM asks for the customer's PIN code (4 digits)* *Verify account code and PIN - The account code and PIN are verified to determine if the account is valid and if the PIN entered is the correct PIN for the account. For this flow, the account is a valid account and the PIN is the correct PIN associated with this account.*

	ATM Options - The ATM displays the different alternatives available at this ATM. In this flow, the bank customer always selects "Cash Withdraw." *Enter Amount - The ATM the amount to withdraw. For this flow the customer selects a pre-set amount ($10, $20, $50, or $100).* *Authorization - The ATM initiates the verification process with the Banking System by sending the Card ID, PIN, Amount, and Account information as a transaction. For this flow, the Banking System is online and replies with the authorization to complete the cash withdrawal successfully and updates the account balance accordingly.* *Dispense - The Money is dispensed.* *Return Card - The Bank Card is returned.* *Receipt - The receipt is printed and dispensed. The ATM also updates the internal log accordingly.* *Use Case ends with the ATM in the Ready State.*
Alternate Flow 1 - Not a valid Card	*In Basic Flow Step 2 - Verify Bank Card, if the card is not valid, it is ejected with an appropriate message.*
Alternate Flow 2 - ATM out of Money	*At Basic Flow Step 5 - ATM Options, if the ATM is out of money, the "Cash Withdraw" option will not be available.*
Alternate Flow 3 - Insufficient funds in ATM	*At Basic Flow Step 6 - Enter Amount, if the ATM contains insufficient funds to dispense the requested amount, an appropriate message will be displayed, and rejoins the basic flow at Step 6 - Enter Amount.*
Alternate Flow 4 - Incorrect PIN	*At Basic Flow Step 4 - Verify Account and PIN, the customer has three tries to enter the correct PIN.* *If an incorrect PIN is entered, the ATM displays the appropriate message and if there are still tries remaining, this flow rejoins Basic Flow at Step 3 - Enter PIN.*

	If, on the final try the entered PIN number is incorrect, the card is retained, ATM returns to Ready State, and this use case terminates.
Alternate Flow 5 - No Account	*At Basic Flow Step 4 - Verify Account and PIN, if the Banking system returns a code indicating the account could not be found or is not an account which allows withdrawals, the ATM displays the appropriate message and rejoins the Basic Flow at Step 9 - Return Card.*
Alternate Flow 6 - Insufficient Funds in Account	*At Basic Flow Step 7 - Authorization, the Banking system returns a code indicating the account balance is less than the amount entered in Basic Flow Step 6 - Enter Amount, the ATM displays the appropriate message and rejoins the Basic Flow at Step 6 - Enter Amount.*
Alternate Flow 7 - Daily maximum withdrawal amount reached	*At Basic Flow Step 6 - Authorization, the Banking system returns a code indicating that, including this request for withdrawal, the customer has or will have exceeded the maximum amount allowed in a 24 hour period, the ATM displays the appropriate message and rejoins the Basic Flow at Step 6 - Enter Amount.*
Alternate Flow x - Log Error	*If at the Basic Flow Step 10 - Receipt, the log cannot be updated, the ATM enters the "secure mode" in which all functions are suspended. An appropriate alarm is sent to the Bank System to indicate the ATM has suspended operation.*
Alternate Flow y – Quit	*The customer can, at any time, decide to terminate the transaction (quit). The transaction is stopped and the card ejected.*
Alternate Flow z - "Tilt"	*The ATM contains numerous sensors, which monitor different functions, such as power, pressure, exerted on the various doors and gates, and motion detectors. If at any time a sensor is activated, an alarm signal is sent to the Police and the ATM enters a "secure mode" in which all functions are suspended until the appropriate re-start / re-initialize actions are taken.*

In the first iteration, according to the iteration plan, we need to verify that the Cash Withdrawal use case has been implemented correctly. The whole use case has not yet been implemented, only the following flows have been implemented:

Basic Flow - Withdrawal of a pre-set amount ($10, $20, $50, $100)

Alternate Flow 2 - ATM out of Money

Alternate Flow 3 - Insufficient funds in ATM

Alternate Flow 4 - Incorrect PIN

Alternate Flow 5 - No Account / Incorrect Account Type

Alternate Flow 6 - Insufficient funds in Account

In the table below, "n/a" indicates that this condition is not applicable to the test case.

TC ID#	Scenario / Condition	Expected Result
CW1.	*Scenario 1 - Successful Cash Withdraw*	*Successful cash withdrawal.*
CW2.	*Scenario 2 - ATM out of Money*	*Cash Withdraw option unavailable, end of use case*
CW3.	*Scenario 3 - Insufficient funds in ATM*	*Warning message, return to Basic Flow Step 6 - Enter Amount*
CW4.	*Scenario 4 - Incorrect PIN (> 1 left)*	*Warning message, return to Basic Flow Step 4, Enter PIN*
CW5.	*Scenario 4 - Incorrect PIN (= 1 try left)*	*Warning message, return to Basic Flow Step 4, Enter PIN*
CW6.	*Scenario 4 - Incorrect PIN (= 0 tries left)*	*Warning message, card retained, end of use case*

4.3 While Box test design

- The basic coverage measure is:

$$\text{Coverage} = \frac{\text{Number of coverage items exercised}}{\text{Total number of coverage items}} \times 100\%$$

Tested coverage = Số TC (pass, fail)/ Tổng số TC

Tested successful coverage = Số TC pass/ Tổng số TC test – N/A

Statements coverage = Số dòng code được test đến/ tổng số dòng code trong software

- Coverage item: whatever we have been able to count and see whether a test has exercised or used this item - Yếu tố được bao phủ: bất cứ điều gì chúng ta đã có thể đếm và xem liệu một test đã thực hiện hay test đến item này hay chưa

4.3.1 Statement coverage

- Percentage of executable statements exercised by a test suite – Phần tram sô dòng code được thực hiện test bởi 1 bộ test
  - Formula = number of statements exercised / Total number of statements
- Example:
  - Program has 100 statements
  - Test exercise 87 statements
  - Statement coverage =87%
- Statement coverage is normally measured by a software tool. – Độ bao phủ code thường được đo bằng một công cụ phần mềm.
- Typical ad hoc testing achieves 60-75% - Thông thường các dự án chỉ đạt được 60- 75%

Example of statement coverage

Read (a)

If a>6 then

A=b

endif

We need only 1 test case a=7 to have 100% statement coverage

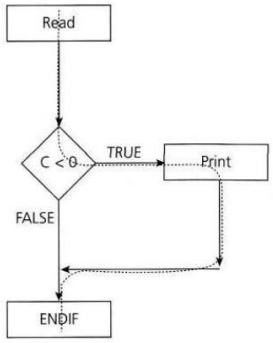
4.3.2 Decision coverage (Branch coverage)

- Percentage of decision outcomes exercised by a test suite - Tỷ lệ các kết quả quyết định được thực hiện bởi một bộ test

Formula = number of decisions outcomes exercised / total number of decision outcomes

- Typical ad hoc testing achieves 40- 60% - Tỷ lệ thông thường 40-60%
- Decision testing is a form of control flow testing
- 100% decision coverage => 100% statement coverage

```
1  READ A
2  READ B
3  C = A - 2 * B
4  IF C <0 THEN
5        PRINT "C negative"
6  ENDIF
```

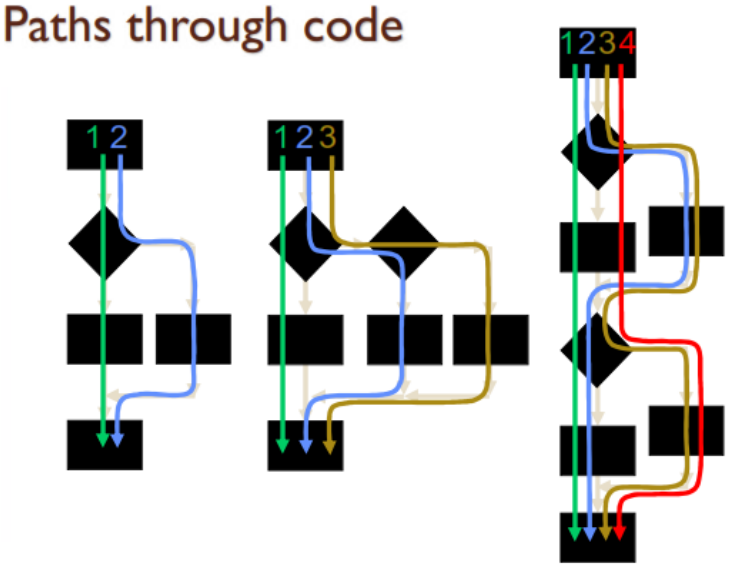


- Case 1: A=20, B-15

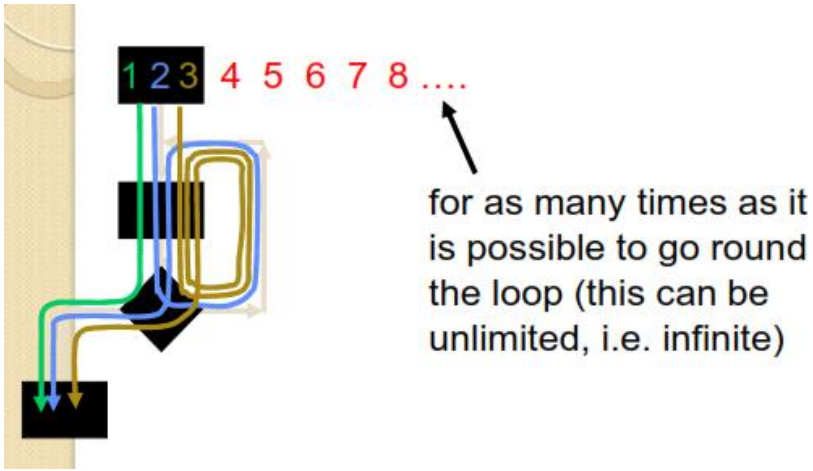
⇒False outcome isn't covered

⇒Add case2: A=10, B=2

4.3.3 Path coverage



Paths through code with loops



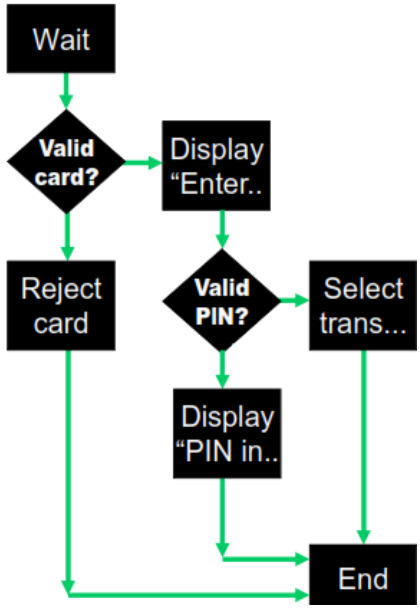
Practices:

Dựa vào sơ đồ sau hãy tính:

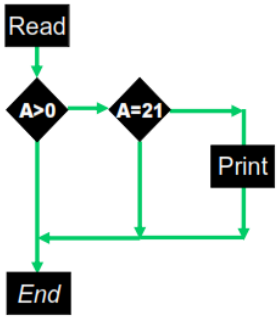
1. Cyclomatic complecity
2. Số TC để đạt 100% Statement coverage
3. Số TC để đạt 100% Branch coverage
4. Số TC để đạt 100% Path through coverage

Example 1

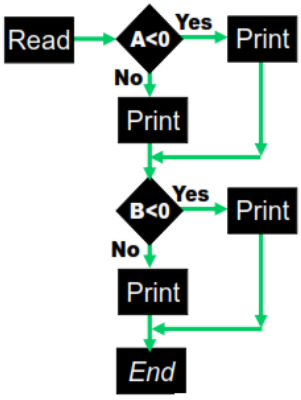
Wait for card to be inserted
IF card is a valid card THEN
  display "Enter PIN number"
  IF PIN is valid THEN
    select transaction
  ELSE (otherwise)
    display "PIN invalid"
ELSE (otherwise)
  reject card
End



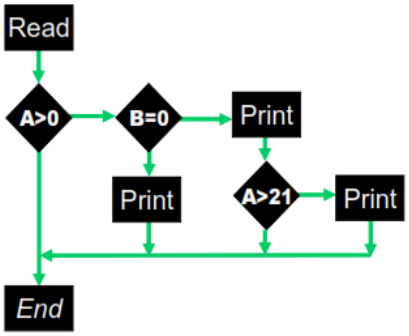
Example 2



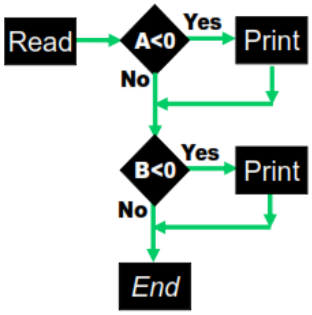
Example 4



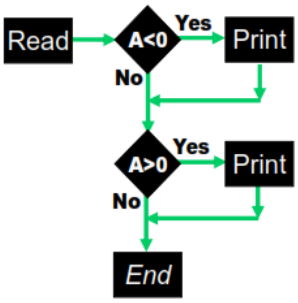
Example 3



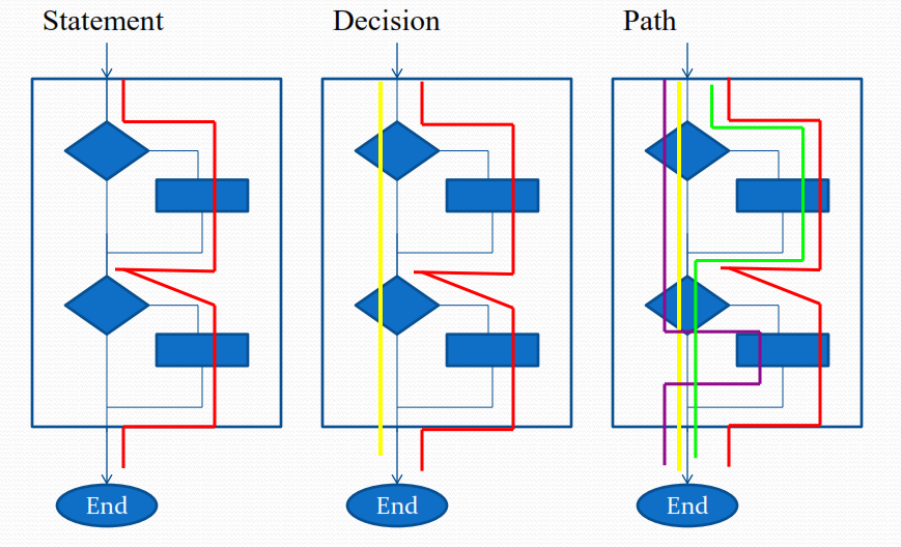
Example 5

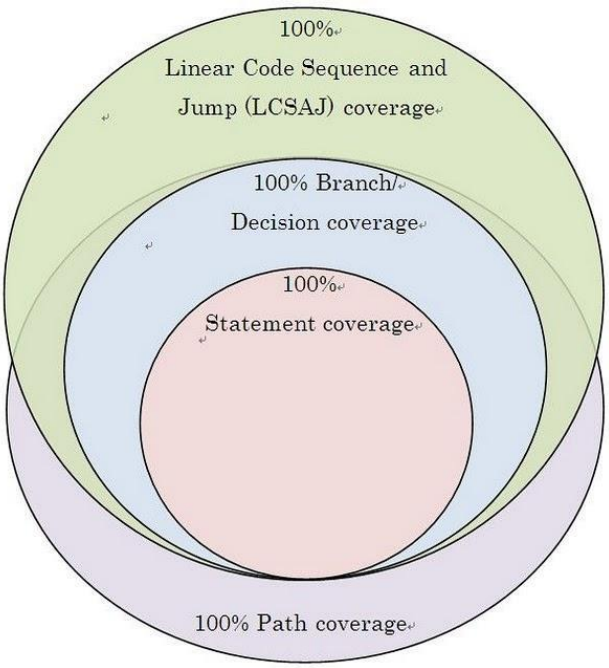


Example 6



Summary while box test





4.4 Experience_base techniques (K2)- Kỹ thuật Experience_base

4.4.1 Error Guessing - Đoán lỗi

Error guessing is a technique used to anticipate the occurrence of mistakes, defects, and failures, based on the tester's knowledge, including - Đoán lỗi là một kỹ thuật được sử dụng để dự đoán sự xuất hiện của sai lầm, khuyết điểm và failure, dựa trên kiến thức của tester, bao gồm:

- How the application has worked in the past - Ứng dụng đã hoạt động như thế nào trong quá khứ
- What types of mistakes the developers tend to make - Những loại sai lầm mà các developer có xu hướng mắc phải
- Failures that have occurred in other applications - Lỗi đã xảy ra trong các ứng dụng khác

- Experience about defect and failure data - Kinh nghiệm về dữ liệu lỗi và lỗi
- Common knowledge about why software fails - Kiến thức phổ biến về lý do tại sao phần mềm bị lỗi

Error guessing technique is to create a list of possible mistakes, defects, and failures, and design tests that will expose those failures and the defects that caused them - Kỹ thuật đoán lỗi là tạo ra một danh sách các lỗi, lỗi và lỗi có thể xảy ra, và các test thiết kế sẽ phơi bày những lỗi đó và các defect gây ra chúng.

4.4.2 Exploratory Testing

In exploratory testing, informal tests are designed, executed, logged, and evaluated dynamically during test execution. The test results are used to learn more about the component or system, and to create tests for the areas that may need more testing- Trong test thăm dò, các test không được thiết kế testcase, thực hiện, ghi nhật ký và đánh giá linh hoạt trong quá trình thực hiện chạy test. Các kết quả test được sử dụng để tìm hiểu thêm về thành phần hoặc hệ thống và để tạo các test cho các khu vực có thể cần test nhiều hơn

Exploratory testing is sometimes conducted using session-based testing to structure the activity- Test thăm dò đôi khi được thực hiện bằng cách sử dụng theo phương pháp session-based:

- Tester uses a test charter containing test objectives within a defined time-box to guide the testing-Tester sử dụng điều lệ test có chứa các mục tiêu test trong một khung thời gian xác định để hướng dẫn test..
- Tester may use test session sheets to document the steps followed and the discoveries made- Tester có thể sử dụng sheets test để ghi lại các bước tiếp theo và những khám phá được thực hiện.

Exploratory testing is most useful when there are few or inadequate specifications or significant time pressure on testing. Exploratory testing is also useful to complement other more formal testing techniques- Test thăm dò là hữu ích nhất khi có ít hoặc không đủ tài liệu hoặc áp lực thời gian. Test thăm dò cũng hữu ích để bổ sung cho các kỹ thuật test chính thức khác

4.4.3 Checklist-based Testing

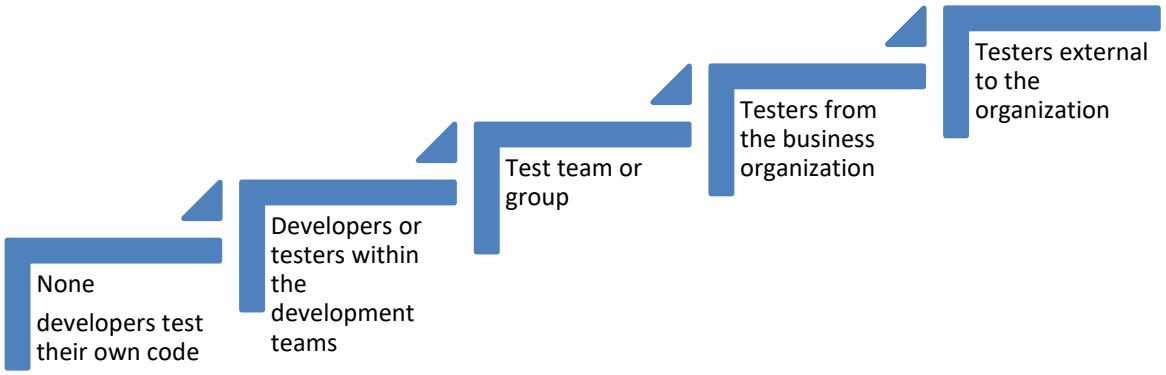
Checklists can be built based on experience, knowledge about what is important for the user, or an understanding of why and how software fails- checklist có thể được xây dựng dựa trên kinh nghiệm, kiến thức về những gì quan trọng đối với người dùng hoặc sự hiểu biết về lý do và cách thức phần mềm bị lỗi.

Checklists can be created to support various test types, including functional and non-functional testing- Checklist có thể được tạo để hỗ trợ các loại test khác nhau, bao gồm test chức năng và phi chức năng.

In the absence of detailed test cases, checklist-based testing can provide guidelines and a degree of consistency - Trong trường hợp không có trường hợp test chi tiết, test dựa trên checklist có thể cung cấp hướng dẫn và mức độ nhất quán

Chapter 5: Test Management

5.1 Test Organization



Benefits of test independence include-Lợi ích của tính độc lập test bao gồm:

- Recognize different kinds of failures - Nhận ra các loại failure khác nhau
- Verify, challenge, or disprove assumptions - Xác minh, test hoặc bác bỏ các giả định

Drawbacks of test independence include- Hạn chế của tính độc lập test bao gồm:

- Isolation from the development team, leading to a lack of collaboration, delays in providing feedback to the development team, or an adversarial relationship with the development team - Cô lập khỏi nhóm phát triển, dẫn đến thiếu sự hợp tác, chậm trễ trong việc cung cấp phản hồi cho nhóm phát triển hoặc mối quan hệ bất lợi với nhóm phát triển
- Developers may lose a sense of responsibility for quality - Các developer có thể mất ý thức trách nhiệm về chất lượng
- Independent testers may be seen as a bottleneck or blamed for delays in release - Tester độc lập có thể được coi là một nút cổ chai hoặc đổ lỗi cho sự chậm trễ trong việc phát hành
- Independent testers may lack some important information (e.g., about the test object) - Tester độc lập có thể thiếu một số thông tin quan trọng (ví dụ: về đối tượng test)

5.1.2 Tasks of a Test Manager and Tester

Testing phase	Test Manager tasks	Tester tasks
Planning	- Plan the test activities - Write and update the test plan(s) - Coordinate the test plan(s) with project managers, product owners, and others - Share testing perspectives with other	- Review and contribute to test plans - Create the detailed test execution schedule

	project activities, such as integration planning	
Analysis and design Implement and execution	- Initiate the analysis, design, implementation, and execution of tests - Support the selection and implementation of tools to support the test process - Support setting up the defect management system - Set up adequate configuration management of testware - Decide about the implementation of test environment(s)	- Analyze, review, and assess requirements for testability (i.e., the test basis) - Identify and document test conditions, capture traceability - Design, set up, and verify test environment(s) - Design and implement test cases and test procedures - Prepare and acquire test data - Execute tests, evaluate the results - Automate tests as needed - Evaluate non-functional characteristics - Review tests developed by others
Monitoring and control	- Monitor test progress and results, and check the status of exit criteria - Create test progress reports and test summary reports - Adapt planning based on test results and progress - Take any actions necessary for test control	- Use appropriate tools to facilitate the test process
Organization task	- Develop or review a test policy and test strategy - Introduce suitable metrics for measuring test progress and evaluating the quality of the testing and the product - Promote and advocate the testers, the test team - Develop the skills and careers of	-

	testers	

Testing phase	Nhiệm vụ quản lý test	Nhiệm vụ test viên
Planning	- Lập kế hoạch hoạt động test - Viết và cập nhật (các) kế hoạch test - Phối hợp (các) kế hoạch test với người quản lý dự án, chủ sở hữu sản phẩm và những người khác - Chia sẻ quan điểm test với các hoạt động khác của dự án, chẳng hạn như lập kế hoạch tích hợp	- Xem xét và đóng góp cho kế hoạch test - Tạo lịch thực hiện test chi tiết
Analysis and design Implement and execution	- Khởi tạo phân tích, thiết kế, chuẩn bị và thực hiện test - Lựa chọn và hỗ trợ thực hiện các công cụ để hỗ trợ quá trình test - Hỗ trợ thiết lập hệ thống quản lý lỗi - Thiết lập quản lý cấu hình đầy đủ của phần mềm test - Quyết định về việc thực hiện (các) môi trường test	- Phân tích, xem xét và đánh giá các yêu cầu về khả năng test - Xác định và ghi lại các điều kiện test, truy xuất nguồn gốc - Thiết kế, thiết lập và xác minh (các) môi trường test - Thiết kế và thực hiện các trường hợp test và quy trình test - Chuẩn bị và thu thập dữ liệu test - Thực hiện test, đánh giá kết quả - Tự động test khi cần thiết - Đánh giá các đặc điểm phi chức năng - Xem xét các bài test được phát triển bởi những người khác
Monitoring and control	- Theo dõi tiến độ và kết quả test, và test trạng thái của tiêu chí thoát - Tạo báo cáo tiến độ test và báo cáo tóm tắt test - Lập kế hoạch thích ứng dựa trên kết quả và tiến độ test - Thực hiện bất kỳ hành động cần thiết	- Sử dụng các công cụ thích hợp để tạo thuận lợi cho quá trình test

	để kiểm soát test	
Organization task	- Phát triển hoặc xem xét một chính sách test và chiến lược test - Giới thiệu các số liệu phù hợp để đo lường tiến độ test và đánh giá chất lượng test và sản phẩm - Thúc đẩy và ủng hộ những tester, nhóm test - Phát triển kỹ năng và nghề nghiệp của tester	-

5.2 Test Planning and Estimation

5.2.1 Purpose and Content of a Test Plan - Mục đích và nội dung của một kế hoạch test

- Determining the scope, objectives, and risks of testing - Xác định phạm vi, mục tiêu và rủi ro của test
- Defining the overall approach of testing - Xác định cách tiếp cận tổng thể của test
- Integrating and coordinating the test activities into the software lifecycle activities - Tích hợp và phối hợp các hoạt động test vào các hoạt động vòng đời phần mềm
- Making decisions about what to test, the people and other resources - Ra quyết định về những gì cần test, người dân và các tài nguyên khác
- Scheduling of test analysis, design, implementation, execution, and evaluation activities - Lập kế hoạch phân tích test, thiết kế, thực hiện, thực hiện và đánh giá các hoạt động
- Selecting metrics for test monitoring and control - Chọn số liệu để theo dõi và kiểm soát test
- Budgeting for the test activities - Ngân sách cho các hoạt động test
- Determining the level of detail and structure for test documentation - Xác định mức độ chi tiết và cấu trúc của tài liệu test

5.2.2 Test Strategy and Test Approach - Chiến lược test và phương pháp test

Analytical: analysis of some factor (e.g., requirement or risk) - phân tích một số yếu tố (ví dụ: yêu cầu hoặc rủi ro)..

Model-Based: tests are designed based on some model of some required aspect of the product. Examples of such models include business process models, state models, and reliability growth models - các test được thiết kế dựa trên một số mô hình về một số khía cạnh bắt buộc của sản phẩm. Ví dụ về các mô hình như vậy bao gồm các mô hình quy trình kinh doanh, mô hình nhà nước và mô hình tăng trưởng độ tin cậy.

Methodical: use of some predefined set of tests or test conditions, such as a taxonomy of common or likely types of failures, a list of important quality characteristics, or standards for mobile apps or web pages - sử dụng một số bộ test hoặc điều kiện test được xác định trước, chẳng hạn như phân loại các loại lỗi phổ biến hoặc có khả năng, danh sách các đặc điểm chất lượng quan trọng hoặc tiêu chuẩn cho ứng dụng di động hoặc trang web

Process-compliant (or standard-compliant): based on external rules and standards, process - dựa trên các quy tắc và tiêu chuẩn bên ngoài, quy trình

Directed (or consultative): base on the advice, guidance, or instructions of stakeholders, business domain experts, or technology experts - dựa trên lời khuyên, hướng dẫn hoặc hướng dẫn của các bên liên quan, chuyên gia trong lĩnh vực kinh doanh hoặc chuyên gia công nghệ

Regression-averse: highly automation of regression tests, and standard test suites - tự động hóa cao các bài test hồi quy và bộ test tiêu chuẩn.

Reactive: Tests are designed and implemented, and may immediately be executed in response to knowledge gained from prior test results. Exploratory testing is a common technique employed in reactive strategies - Các test được thiết kế và thực hiện, và có thể được thực hiện ngay lập tức để đáp ứng với kiến thức thu được từ các kết quả test trước đó. Test thăm dò là một kỹ thuật phổ biến được sử dụng trong các chiến lược phản ứng.

5.2.3 Entry Criteria and Exit Criteria (Definition of Ready and Definition of Done) - Entry Criteria and Exit Criteria (Definition of Ready and Definition of Done)

Typical entry criteria include- Tiêu chí đầu vào điển hình bao gồm::

- Availability of testable requirements, user stories, and/or models - Tính khả dụng của các yêu cầu có thể test, yêu cầu của người dùng và / hoặc mô hình
- Availability of test items that have met the exit criteria for any prior test levels - Tính khả dụng của các mục test đã đáp ứng tiêu chí thoát cho mọi test level trước đó
- Availability of test environment - Sự sẵn có của môi trường test
- Availability of necessary test tools - Có sẵn các công cụ test cần thiết
- Availability of test data and other necessary resources - Có sẵn dữ liệu test và các tài nguyên cần thiết khác

Typical exit criteria include - Tiêu chí kết thúc điển hình bao gồm:

- Planned tests have been executed -Test theo kế hoạch đã được thực hiện
- A defined level of coverage - Một mức độ bao phủ xác định
- The number of unresolved defects is within an agreed limit - Số lượng defect chưa được giải quyết nằm trong giới hạn đã thỏa thuận
- The number of estimated remaining defects is sufficiently low - Số lượng defect còn lại ước tính là đủ thấp

- The evaluated levels of reliability, performance efficiency, usability, security, and other relevant quality characteristics are sufficient - Các mức đánh giá độ tin cậy, hiệu quả hoạt động, khả năng sử dụng, bảo mật và các đặc tính chất lượng khác có liên quan là đủ

5.2.4 Test Execution Schedule - Lịch trình thực hiện

Once the various test cases and test procedures are produced and assembled into test suites - Khi các trường hợp test và quy trình test khác nhau được sản xuất và lắp ráp thành các bộ test

The test suites can be arranged in a test execution schedule that defines the order in which they are to be run - Các bộ test có thể được sắp xếp trong một lịch trình thực hiện test xác định thứ tự mà chúng sẽ được chạy.

The test execution schedule should take into account such factors as prioritization, dependencies, confirmation tests, regression tests, and the most efficient sequence for executing the tests - Lịch thực hiện test cần tính đến các yếu tố như ưu tiên, phụ thuộc, test xác nhận, test hồi quy và trình tự hiệu quả nhất để thực hiện các test.

5.2.5 Factors Influencing the Test Effort - Các yếu tố ảnh hưởng đến nỗ lực test

Test effort estimation involves predicting the amount of test-related work that will be needed in order to meet the objectives of the testing for a particular project, release, or iteration - Ước tính nỗ lực test bao gồm dự đoán số lượng công việc liên quan đến test sẽ cần để đáp ứng các mục tiêu của test cho một dự án cụ thể, phát hành hoặc lặp lại

Product characteristics- Đặc tính sản phẩm

- The risks associated with the product - Những rủi ro liên quan đến sản phẩm
- The quality of the test basis - Chất lượng của cơ sở test
- The size of the product - Kích thước của sản phẩm
- The complexity of the product domain - Sự phức tạp của miền sản phẩm
- The requirements for quality characteristics (e.g., security, reliability) - Các yêu cầu về đặc tính chất lượng (ví dụ: bảo mật, độ tin cậy)
- The required level of detail for test documentation - Mức độ chi tiết cần thiết cho tài liệu test
- Requirements for legal and regulatory compliance - Yêu cầu tuân thủ pháp luật và quy định

Development process characteristics - Đặc điểm quá trình phát triển

- The stability and maturity of the organization - Sự ổn định và trưởng thành của tổ chức
- The development model in use - Mô hình phát triển được sử dụng

- The test approach - Phương pháp test
- The tools used - Các công cụ được sử dụng
- The test process - Quá trình test
- Time pressure - Áp lực thời gian

People characteristics - Đặc điểm con người

- The skills and experience of the people involved, especially with similar projects and products (e.g., domain knowledge) - Kỹ năng và kinh nghiệm của những người liên quan, đặc biệt là với các dự án và sản phẩm tương tự (ví dụ: kiến thức về tên miền)
- Team cohesion and leadership - Sự gắn kết và lãnh đạo nhóm

Test results - Kết quả test

- The number and severity of defects found -Số lượng và mức độ nghiêm trọng của defect được tìm thấy
- The amount of rework required - Số lượng làm lại cần thiết

5.2.6 Test Estimation Techniques - Kỹ thuật ước tính test

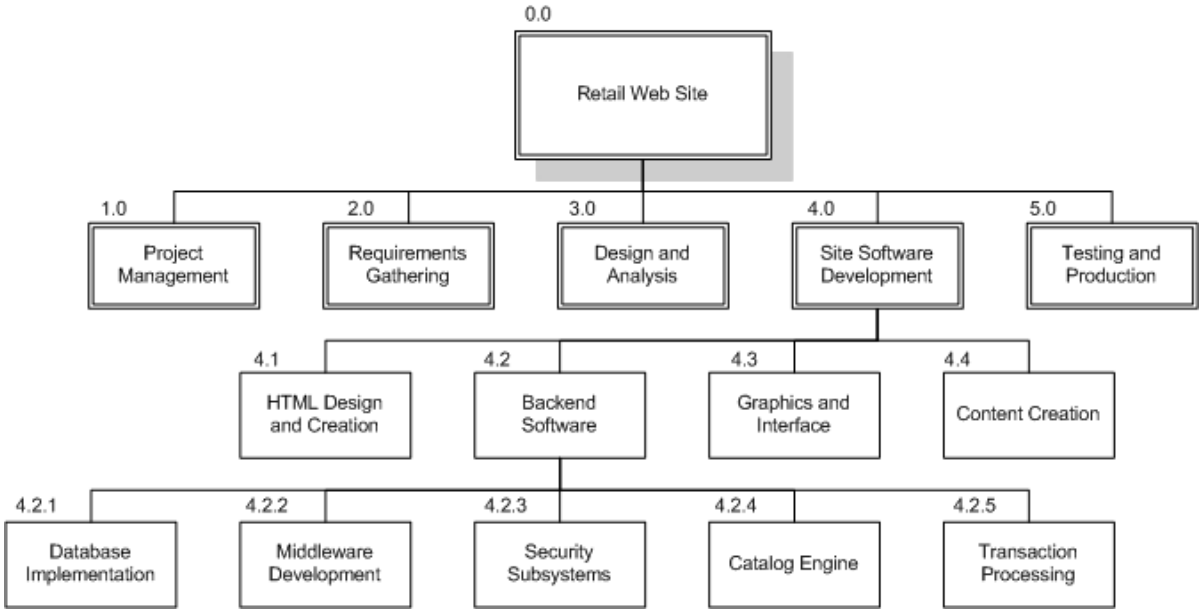
Two of the most commonly used techniques are - Hai trong số các kỹ thuật được sử dụng phổ biến nhất là::

1. The metrics-based technique: estimating the test effort based on metrics of former similar projects, or based on typical values - Kỹ thuật dựa trên số liệu: ước tính nỗ lực test dựa trên số liệu của các dự án tương tự trước đây hoặc dựa trên các giá trị tiêu biểu

	Project management	Requirement development	Design	Coding	Testing	Quality assurance	CM	Risk buffer
Average 100 old projects	10%	15%	10%	25%	35%	2.5%	2.5%	10%
Project A	?	?	?	1250pd	??	?	?	

2. The expert-based technique: estimating the test effort based on the experience of the owners of the testing tasks or by experts - Kỹ thuật dựa trên chuyên gia: ước tính nỗ lực test dựa trên kinh nghiệm của chủ sở hữu các nhiệm vụ test hoặc bởi các chuyên gia

WBS- Work Breakdown Structure



5.3 Test Monitoring and Control - Test giám sát và kiểm soát

Test monitoring is to gather information and provide feedback and visibility about test activities, measure and assess test progress and measure whether the test exit criteria - Giám sát test là thu thập thông tin và cung cấp phản hồi và khả năng hiển thị về các hoạt động test, đo lường và đánh giá tiến trình test và đo lường xem các tiêu chí thoát test

Test control describes any guiding or corrective actions taken - Kiểm soát test mô tả bất kỳ hành động hướng dẫn hoặc khắc phục được thực hiện:

- Re-prioritizing tests when an identified risk occurs (e.g., software delivered late) - Ưu tiên test lại khi xảy ra rủi ro đã xác định (ví dụ: phần mềm được giao trễ)
- Changing the test schedule due to availability or unavailability of a test environment or other resources - Thay đổi lịch test do tính khả dụng hoặc không có của môi trường test hoặc các tài nguyên khác
- Re-evaluating whether a test item meets an entry or exit criterion due to rework - Đánh giá lại liệu một mục test có đáp ứng tiêu chí xuất nhập cảnh hay không do làm lại

5.3.1 Metrics Used in Testing - Số liệu được sử dụng trong test

Metrics can be collected during and at the end of test activities in order to assess - Số liệu có thể được thu thập trong và khi kết thúc các hoạt động test để đánh giá:

- Progress against the planned schedule and budget - Tiến độ so với kế hoạch và ngân sách
- Current quality of the test object - Chất lượng hiện tại của đối tượng test
- Adequacy of the test approach - Sự phù hợp của phương pháp test

- Effectiveness of the test activities with respect to the objectives - Hiệu quả của các hoạt động test đối với các mục tiêu

Common test metrics include - Các số liệu test phổ biến bao gồm:

- Percentage of planned work done in each test activities - Tỷ lệ phần trăm công việc được lên kế hoạch thực hiện trong mỗi hoạt động test
- Test case execution (e.g., number of test cases run/not run, test cases passed/failed, and/or test conditions passed/failed) - Thực hiện trường hợp test (ví dụ: số trường hợp test chạy / không chạy, trường hợp test đã vượt qua / không thành công và / hoặc điều kiện test đã vượt qua / failure)
- Defect information (e.g., defect density, defects found and fixed, failure rate, and confirmation test results) - Thông tin lỗi (ví dụ: mật độ lỗi, lỗi được tìm thấy và cố định, tỷ lệ failure và kết quả test xác nhận)
- Test coverage of requirements, user stories, acceptance criteria, risks, or code - Test phạm vi yêu cầu, yêu cầu của người dùng, tiêu chí chấp nhận, rủi ro hoặc code
- Task completion, resource allocation and usage, and effort - Hoàn thành nhiệm vụ, phân bổ và sử dụng tài nguyên, và nỗ lực
- Cost of testing, including the cost compared to the benefit of finding the next defect or the cost compared to the benefit of running the next test - Chi phí test, bao gồm chi phí so với lợi ích của việc tìm ra defect tiếp theo hoặc chi phí so với lợi ích của việc chạy test tiếp theo

5.3.2 Purposes, Contents, and Audiences for Test Reports - Mục đích, nội dung và đối tượng cho báo cáo test

The purpose of test reporting is to summarize and communicate test activity information, both during and at the end of a test activity - Mục đích của báo cáo test là tóm tắt và truyền đạt thông tin hoạt động test, cả trong và khi kết thúc hoạt động test

Content common to test progress reports - Nội dung phổ biến để test báo cáo tiến độ::

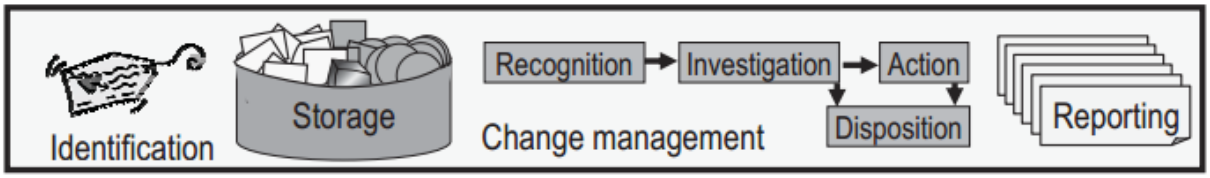
- The status of the test activities and progress against the test plan - Tình trạng của các hoạt động test và tiến độ so với kế hoạch test
- Factors impeding progress - Các yếu tố cản trở tiến độ
- Testing planned for the next reporting period - Test kế hoạch cho giai đoạn báo cáo tiếp theo
- The quality of the test object - Chất lượng của đối tượng test

Typical test progress reports and test summary reports may include - Báo cáo tiến độ test điển hình và báo cáo tóm tắt test có thể bao gồm::

- Summary of testing performed - Tóm tắt test thực hiện
- Analysis the information on what occurred during a test period - Phân tích thông tin về những gì đã xảy ra trong một giai đoạn test
- (Variances) Deviations from plan - Sai lệch so với kế hoạch

- Metrics of defects, test cases, test coverage, activity progress, and resource consumption - Số liệu về lỗi, trường hợp test, phạm vi test, tiến độ hoạt động và mức tiêu thụ tài nguyên.
- (Evaluation) Residual risks - Rủi ro tồn tại
- Reusable test work products produced - Sản phẩm làm việc test tái sản xuất

5.4 Configuration Management - Quản lý cấu hình



The purpose of configuration management is to establish and maintain the integrity of the component or system, the testware, and their relationships to one another through the project and product lifecycle - Mục đích của quản lý cấu hình là thiết lập và duy trì tính toàn vẹn của thành phần hoặc hệ thống, phần mềm test và mối quan hệ của chúng với nhau thông qua vòng đời dự án và sản phẩm::

- All test items are uniquely identified, version controlled, tracked for changes, and related to each other - Tất cả các mục test được xác định duy nhất, kiểm soát phiên bản, theo dõi các thay đổi và liên quan đến nhau
- All items of testware are uniquely identified, version controlled, tracked for changes, related to each other and related to versions of the test item(s) so that traceability can be maintained throughout the test process - Tất cả các mục của phần mềm test được xác định duy nhất, kiểm soát phiên bản, theo dõi các thay đổi, liên quan đến nhau và liên quan đến các phiên bản của (các) mục test để có thể duy trì khả năng truy nguyên trong suốt quá trình test
- All identified documents and software items are referenced unambiguously in test documentation - Tất cả các tài liệu được xác định và các mục phần mềm được tham chiếu rõ ràng trong tài liệu test

5.5 Risks and Testing - Rủi ro và test

5.5.1 Definition of Risk - Định nghĩa rủi ro

Risk involves the possibility of an event in the future which has negative consequences - Rủi ro liên quan đến khả năng của một sự kiện trong tương lai có hậu quả tiêu cực.

The level of risk is determined by the likelihood of the event and the impact (the harm) from that event - Mức độ rủi ro được xác định bởi khả năng của sự kiện và tác động (tác hại) từ sự kiện đó.

5.5.2 Product and Project Risks - Rủi ro sản phẩm và dự án

Project risk	Product risk (quality risks)
A risk to the project's capability to deliver products: Scope, cost, Time	A risk to quality of product
Related to management and control of the (test) project: - Skill, training and staff shortages - Personnel issues	Directly related to the test object - Failure-prone software delivered - The potential that the software/hardware could cause harm to an individual or company

- Political issues - Technical issues - Schedules - Budget - Work product: SRS, code, design, test documents	- Poor software characteristics (e.g., functionality, reliability, usability and performance) - Poor data integrity and quality - Software that does not perform its intended functions - User experience (UX) feedback might not meet product expectations - A loop control structure may be coded incorrectly - A particular computation may be performed incorrectly in some circumstances
Actions: PM and Test manager - Mitigation or reduce risk	Actions: Tester - Risk-based testing

Project risk	**Product risk (quality risks)**
Rủi ro đối với khả năng cung cấp sản phẩm của dự án: Phạm vi, chi phí, Thời gian	Nguy cơ đối với chất lượng sản phẩm
Liên quan đến quản lý và kiểm soát dự án (test): - Kỹ năng, đào tạo và thiếu hụt nhân viên - Vấn đề nhân sự - Vấn đề chính trị - Vấn đề kỹ thuật - Lịch trình - Ngân sách -	Liên quan trực tiếp đến đối tượng test - Phần mềm dễ bị lỗi - Khả năng phần mềm / phần cứng có thể gây hại cho cá nhân hoặc công ty - Đặc điểm phần mềm kém (ví dụ: chức năng, độ tin cậy, khả năng sử dụng và hiệu suất) - Tính toàn vẹn và chất lượng dữ liệu kém - Phần mềm không thực hiện các chức năng dự định của nó - Phản hồi về trải nghiệm người dùng (UX) có thể không đáp ứng mong đợi của sản phẩm - Một cấu trúc điều khiển vòng lặp có thể được code không chính xác - Một tính toán cụ thể có thể được thực hiện không chính xác trong một số trường hợp -

5.5.3 Risk-based Testing and Product Quality - Test dựa trên rủi ro và chất lượng sản phẩm

Risk is used to focus the effort required during testing. It is used to decide where and when to start testing and to identify areas that need more attention - Risks are used to focus the effort required during testing. It is used to decide where and when to start testing and to identify areas that need more attention - Rủi ro được sử dụng để tập trung nỗ lực cần thiết trong quá trình test. Nó được sử dụng để quyết định nơi và khi nào bắt đầu test và để xác định các khu vực cần chú ý hơn.

Testing is used to reduce the probability of an adverse event occurring, or to reduce the impact of an adverse event - Test được sử dụng để giảm xác suất của các sự kiện bất lợi hoặc để giảm tác động của các sự kiện bất lợi.

Testing is used as a risk mitigation activity, to provide feedback about identified risks, as well as providing feedback on residual (unresolved) risks - Test được sử dụng như một hoạt động giảm thiểu rủi ro, để cung cấp phản hồi về các rủi ro đã xác định, cũng như cung cấp phản hồi về các rủi ro còn lại (chưa được giải quyết).

In a risk-based approach, the results of product risk analysis are used to - Theo cách tiếp cận dựa trên rủi ro, kết quả phân tích rủi ro sản phẩm được sử dụng để:

- Determine the test techniques to be employed - Xác định kỹ thuật test nào sẽ được sử dụng
- Determine the particular levels and types of testing to be performed (e.g., security testing, accessibility testing) - Xác định các cấp độ và loại test cụ thể sẽ được thực hiện (ví dụ: test bảo mật, test khả năng truy cập)
- Determine the extent of testing to be carried out - Xác định phạm vi test sẽ được thực hiện
- Prioritize testing in an attempt to find the critical defects as early as possible - Test ưu tiên trong nỗ lực tìm kiếm các defect nghiêm trọng càng sớm càng tốt
- Determine whether any activities in addition to testing could be employed to reduce risk (e.g., providing training to inexperienced designers) - Xác định xem có hoạt động nào khác ngoài test có thể được sử dụng để giảm rủi ro hay không (ví dụ: cung cấp đào tạo cho các nhà thiết kế thiếu kinh nghiệm)

Risk activities include - Các hoạt động rủi ro bao gồm:

- Analyze (and re-evaluate on a regular basis) what can go wrong (risks) - Phân tích (và đánh giá lại một cách thường xuyên) những gì có thể sai (rủi ro)
- Determine which risks are important to deal with (risk level: likelihood, impact) - Xác định rủi ro nào là quan trọng để giải quyết (mức độ rủi ro: khả năng, tác động)
- Implement actions to mitigate those risks - Hãy hành động để giảm thiểu những rủi ro đó
- Make contingency plans to deal with the risks should they become actual events - Lập kế hoạch dự phòng để đối phó với rủi ro nếu chúng trở thành sự kiện thực sự

The testing may identify new risks, help to determine what risks should be mitigated, and lower uncertainty about risks - Test có thể xác định rủi ro mới, giúp xác định rủi ro nào cần được giảm thiểu và giảm rủi ro không chắc chắn

5.6 Defect Management

Since one of the objectives of testing is to find defects, the discrepancies between actual and expected outcomes need to be logged as incidents. Mục tiêu của test là tìm lỗi, vì thế tất cả những sự sai lệch giữa kết quả thực tế và kết quả mong đợi đều phải được ghi nhận là 1 incidents

An incident must be investigated and may turn out to be a defect. Appropriate actions to dispose incidents and defects should be defined. Incidents and defects should be tracked from discovery and classification to correction and confirmation of the solution. – Một sự cố phải được truy vết và có thể chuyển thành lỗi.

Các hành động thích hợp cần được xác định để giải quyết sự cố và lỗi. Sự cố và lỗi nên được theo dõi từ khi phát hiện ra và phân loại để sửa chữa

In order to manage all incidents to completion, an organization should establish an incident management process and rules for classification. – Để quản lý được tất cả các sự cố đến khi hoàn toàn, tổ chức nên thiết lập quy trình quản lý sự cố, các quy tắc phân loại

Incident reports have the following objectives- Báo cáo sự cố có các mục đích sau:

- Provide developers and other parties with feedback about the problem to enable identification, isolation and correction as necessary – Cung cấp cho LTV và các bên thông tin phản hồi về sự cố để có thể xác định, định vị nó và sửa chữa nó nếu cần
- Provide test leaders a means of tracking the quality of the system under test and the progress of the testing – Cung cấp cho các test leader một phương tiện để theo dõi chất lượng
- Provide ideas for test process improvement – Cung cấp ý tưởng cho việc cải tiến quy trình test

A defect report filed during dynamic testing typically includes - Một báo cáo lỗi được nộp trong quá trình test động thường bao gồm:

- An identifier - Một định danh
- A title and a short summary of the defect being reported - Một tiêu đề và một bản tóm tắt ngắn về defect được báo cáo
- Date of the defect report, issuing organization, and author - Ngày báo cáo lỗi, tổ chức phát hành và tác giả
- Identification of the test item (configuration item being tested) and environment - Xác định mục test (mục cấu hình đang được test) và môi trường
- The development lifecycle phase(s) in which the defect was observed - Giai đoạn vòng đời phát triển trong đó quan sát thấy defect
- A description of the defect to enable reproduction and resolution, including logs, database dumps screenshots, or recordings (if found during test execution) - Mô tả về lỗi để cho phép tái tạo và phân giải, bao gồm nhật ký, cơ sở dữ liệu chụp màn hình hoặc bản ghi (nếu được tìm thấy trong khi thực hiện test)
- Expected and actual results - Kết quả mong đợi và thực tế
- Scope or degree of impact (severity) of the defect on the interests of stakeholder(s) - Phạm vi hoặc mức độ ảnh hưởng (mức độ nghiêm trọng) của defect đối với lợi ích của (các) bên liên quan
- Urgency/priority to fix - Sự khẩn cấp / ưu tiên cần khắc phục
- State of the defect report (e.g., open, deferred, duplicate, waiting to be fixed, awaiting confirmation testing, re-opened, closed) - Báo cáo trạng thái lỗi (ví dụ: mở, hoãn, trùng lặp, chờ sửa chữa, chờ test xác nhận, mở lại, đóng)
- Conclusions, recommendations and approvals - Kết luận, khuyến nghị và phê duyệt
- Global issues, such as other areas that may be affected by a change resulting from the defect - Các vấn đề chung, chẳng hạn như các lĩnh vực khác có thể bị ảnh hưởng bởi một thay đổi do defect
- Change history, such as the sequence of actions taken by project team members with respect to the defect to isolate, repair, and confirm it as fixed - Thay đổi lịch sử, chẳng hạn như chuỗi các hành động được thực hiện bởi các thành viên trong nhóm dự án liên quan đến lỗi để cách ly, sửa chữa và xác nhận nó là cố định

- References, including the test case that revealed the problem - Tài liệu tham khảo, bao gồm cả trường hợp test đã tiết lộ vấn đề

Chapter 6: Tool Support for Testing

6.1 Test Tool Considerations - Công cụ test xem xét

6.1.1 Test Tool Classification - Phân loại công cụ test

Test tools can have some purposes - Công cụ test có thể có một số mục đích:

- Improve the efficiency of test activities by automating repetitive tasks or tasks that require significant resources when done manually (e.g., test execution, regression testing) - Cải thiện hiệu quả của các hoạt động test bằng cách tự động hóa các tác vụ lặp đi lặp lại hoặc các tác vụ yêu cầu tài nguyên quan trọng khi được thực hiện thủ công (ví dụ: thực hiện test, test hồi quy)
- Improve the efficiency of test activities by supporting manual test activities throughout the test process - Nâng cao hiệu quả của các hoạt động test bằng cách hỗ trợ các hoạt động test thủ công trong suốt quá trình test
- Improve the quality of test activities by allowing for more consistent testing and a higher level of defect reproducibility - Cải thiện chất lượng của các hoạt động test bằng cách cho phép test phù hợp hơn và mức độ tái tạo defect cao hơn
- Automate activities that cannot be executed manually (e.g., large scale performance testing) - Tự động hóa các hoạt động không thể được thực hiện thủ công (ví dụ: test hiệu suất quy mô lớn)
- Increase reliability of testing (e.g., by automating large data comparisons or simulating behavior) - Tăng độ tin cậy của test (ví dụ: bằng cách tự động so sánh dữ liệu lớn hoặc hành vi mô phỏng)

Tool categories - Các loại công cụ

Support management

Test management tools
Requirements management tools
Defect management tools
Configuration management tools
Continuous integration tools (D)

Support for static testing

Tools that support reviews
Static analysis tools (D)

Support for test execution and logging

Test execution tools
Coverage tools
Test harnesses (D)
Unit test framework tools (D)

Support for performance measurement and dynamic analysis

Performance testing tools
Monitoring tools
Dynamic analysis tools (D)

Support for test design and implementation

Test design tools
Model-Based testing tools
Test data preparation tools
Acceptance test driven development (ATDD) and behavior driven development (BDD) tools
Test driven development (TDD) tools

Support for specialized testing needs

Data quality assessment
Data conversion and migration
Usability testing
Accessibility testing
Localization testing
Security testing
Portability testing

6.1.2 Benefits and Risks of Test Automation - Lợi ích và rủi ro của tự động hóa test

Potential benefits of using tools to support test execution include - Lợi ích tiềm năng của việc sử dụng các công cụ để hỗ trợ thực hiện test bao gồm:

- Reduction in repetitive manual work (e.g., running regression tests, environment set up/tear down tasks, re-entering the same test data, and checking against coding standards), thus saving time - Giảm công việc thủ công lặp đi lặp lại (ví dụ: chạy test hồi quy, thiết lập / xé bỏ các tác vụ môi trường, nhập lại cùng một dữ liệu test và test theo các tiêu chuẩn code), do đó tiết kiệm thời gian
- Greater consistency and repeatability (e.g., test data is created in a coherent manner, tests are executed by a tool in the same order with the same frequency, and tests are consistently derived from requirements) - Tính nhất quán và độ lặp lại cao hơn (ví dụ: dữ liệu test được tạo theo cách mạch lạc, các test được thực hiện bởi một công cụ theo cùng thứ tự với cùng tần số và các test luôn xuất phát từ yêu cầu)
- More objective assessment (e.g., static measures, coverage) - Đánh giá khách quan hơn (ví dụ: các biện pháp tĩnh, bảo hiểm)

- Easier access to information about testing (e.g., statistics and graphs about test progress, defect rates and performance) - Truy cập dễ dàng hơn đến thông tin về test (ví dụ: thống kê và biểu đồ về tiến trình test, tỷ lệ lỗi và hiệu suất)

Potential risks of using tools to support testing include - Rủi ro tiềm tàng của việc sử dụng các công cụ để hỗ trợ test bao gồm:

- Expectations for the tool may be unrealistic (including functionality and ease of use) - Kỳ vọng cho công cụ này có thể không thực tế (bao gồm cả chức năng và dễ sử dụng)
- The time, cost and effort for the initial introduction of a tool may be under-estimated (including training and external expertise) - Thời gian, chi phí và nỗ lực cho việc giới thiệu ban đầu của một công cụ có thể được ước tính thấp (bao gồm đào tạo và chuyên môn bên ngoài)
- The time and effort needed to achieve significant and continuing benefits from the tool may be under-estimated (including the need for changes in the test process and continuous improvement in the way the tool is used) - Thời gian và nỗ lực cần thiết để đạt được lợi ích đáng kể và tiếp tục từ công cụ có thể được ước tính thấp (bao gồm nhu cầu thay đổi trong quy trình test và cải tiến liên tục trong cách sử dụng công cụ)
- The effort required to maintain the test assets generated by the tool may be under-estimated - Nỗ lực cần thiết để duy trì các tài sản test được tạo ra bởi công cụ có thể được ước tính thấp
- The tool may be relied on too much (seen as a replacement for test design or execution, or the use of automated testing where manual testing would be better) - Công cụ này có thể dựa vào quá nhiều (được xem là sự thay thế cho thiết kế test hoặc thực thi hoặc sử dụng test tự động trong đó test thủ công sẽ tốt hơn)
- Version control of test assets may be neglected - Kiểm soát phiên bản của tài sản test có thể bị bỏ qua
- Relationships and interoperability issues between critical tools may be neglected, such as requirements management tools, configuration management tools, defect management tools and tools from multiple vendors - Mối quan hệ và các vấn đề về khả năng tương tác giữa các công cụ quan trọng có thể bị bỏ qua, chẳng hạn như công cụ quản lý yêu cầu, công cụ quản lý cấu hình, công cụ quản lý lỗi và công cụ từ nhiều nhà cung cấp
- The tool vendor may go out of business, retire the tool, or sell the tool to a different vendor

  - Nhà cung cấp công cụ có thể ngừng hoạt động, nghỉ hưu công cụ hoặc bán công cụ cho một nhà cung cấp khác
- The vendor may provide a poor response for support, upgrades, and defect fixes - Nhà cung cấp có thể cung cấp phản hồi kém cho hỗ trợ, nâng cấp và sửa lỗi
- An open source project may be suspended - Một dự án nguồn mở có thể bị đình chỉ
- A new platform or technology may not be supported by the tool - Một nền tảng hoặc công nghệ mới có thể không được công cụ hỗ trợ
- There may be no clear ownership of the tool (e.g., for mentoring, updates, etc.) - Có thể không có quyền sở hữu rõ ràng của công cụ (ví dụ: để cố vấn, cập nhật, v.v.)

6.1.3 Special Considerations for Test Execution and Test Management Tools - Cân nhắc đặc biệt cho các công cụ quản lý test và thực thi test

Test execution tools - Test công cụ thực thi

Data-driven scripting technique	Keyword-driven scripting technique	Model-Based testing (MBT)
Data files store test input and expected results in table or spreadsheet	data files store test input, expected results and keywords in table or spreadsheet	a functional specification to be captured in the form of a model, such as an activity diagram
support capture/playback tools	writing script manually	The MBT tool interprets the model in order to create test case specifications which can then be saved in a test management tool and/or executed by a test execution tool

Kỹ thuật kịch bản dựa trên dữ liệu	Kỹ thuật kịch bản dựa trên từ khóa	Test dựa trên mô hình (MBT)
Tệp dữ liệu lưu trữ đầu vào test và kết quả mong đợi trong bảng hoặc bảng tính	tệp dữ liệu lưu trữ đầu vào test, kết quả dự kiến và từ khóa trong bảng hoặc bảng tính	một đặc tả chức năng được ghi lại dưới dạng mô hình, chẳng hạn như sơ đồ hoạt động
hỗ trợ các công cụ chụp / phát lại	viết kịch bản thủ công	Công cụ MBT diễn giải mô hình để tạo ra các đặc tả trường hợp test, sau đó có thể được lưu trong một công cụ quản lý test và / hoặc được thực thi bởi một công cụ thực thi test

Example keyword driven:

Keyword	User	Password	Result
Add_User	User1	Pass1	User added message
Add_User	@Rec34	@Rec35	User added message
Reset_Password	User1	Welcome	Password reset confirmation message
Delete_User	User1		Invalid username/password message
Add_User	User3	Pass3	User added message
Delete_User	User2		User not found message

Test management tools - Công cụ quản lý test

Test management tools often need to interface with other tools or spreadsheets for various reasons, including - Các công cụ quản lý test thường cần giao diện với các công cụ hoặc bảng tính khác vì nhiều lý do, bao gồm:

- To produce useful information in a format that fits the needs of the organization - Để tạo ra thông tin hữu ích trong một định dạng phù hợp với nhu cầu của tổ chức
- To maintain consistent traceability to requirements in a requirements management tool - Để duy trì khả năng truy nguyên nhất quán theo yêu cầu trong một công cụ quản lý yêu cầu
- To link with test object version information in the configuration management tool - Để liên kết với thông tin phiên bản đối tượng test trong công cụ quản lý cấu hình

6.2 Effective Use of Tools - Sử dụng hiệu quả các công cụ

6.2.1 Main Principles for Tool Selection - Nguyên tắc chính để lựa chọn công cụ

- Assessment of organizational maturity, strengths and weaknesses and identification of opportunities for an improved test process supported by tools - Đánh giá về mức độ trưởng thành của tổ chức, những điểm mạnh và điểm yếu và xác định các cơ hội cho một quá trình test cải thiện được hỗ trợ bởi các công cụ
- Understanding of the technologies used by the test object(s), in order to select a tool that is compatible with that technology- Hiểu biết về các công nghệ được sử dụng bởi (các) đối tượng test, để chọn một công cụ tương thích với công nghệ đó
- The build and continuous integration tools already in use within the organization, in order to ensure tool compatibility and integration - Các công cụ tích hợp và xây dựng liên tục đã được sử dụng trong tổ chức, để đảm bảo khả năng tương thích và tích hợp công cụ
- Evaluation of the tool against clear requirements and objective criteria - Đánh giá công cụ theo các yêu cầu rõ ràng và tiêu chí khách quan
- Consideration of whether or not the tool is available for a free trial period (and for how long) - Xem xét liệu công cụ có sẵn trong thời gian dùng thử miễn phí hay không (và trong bao lâu)
- Evaluation of the vendor (including training, support and commercial aspects) or support for non-commercial (e.g., open source) tools - Đánh giá của nhà cung cấp (bao gồm các khía cạnh đào tạo, hỗ trợ và thương mại) hoặc hỗ trợ cho các công cụ phi thương mại (ví dụ: nguồn mở)
- Identification of internal requirements for coaching and mentoring in the use of the tool - Xác định các yêu cầu nội bộ để huấn luyện và cố vấn trong việc sử dụng công cụ
- Evaluation of training needs, considering the testing (and test automation) skills of those who will be working directly with the tool(s) - Đánh giá nhu cầu đào tạo, xem xét các kỹ năng test (và test tự động hóa) của những người sẽ làm việc trực tiếp với (các) công cụ
- Consideration of pros and cons of various licensing models (e.g., commercial or open source) - Xem xét ưu và nhược điểm của các mô hình cấp phép khác nhau (ví dụ: thương mại hoặc nguồn mở)
- Estimation of a cost-benefit ratio based on a concrete business case (if required) - Ước tính tỷ lệ chi phí - lợi ích dựa trên trường hợp kinh doanh cụ thể (nếu cần)

6.2.2 Pilot Projects for Introducing a Tool into an Organization

- Gaining in-depth knowledge about the tool, understanding both its strengths and weaknesses
- Evaluate how the tool fits with existing processes and practices, and determine what would need to change - Đánh giá cách công cụ phù hợp với quy trình và thực hành hiện có, và xác định những gì cần phải thay đổi
- Decide on standard ways of using, managing, storing and maintaining the tool and the test assets (e.g., deciding on naming conventions for files and tests, creating libraries and defining the modularity of test suites) - Quyết định về tiêu chuẩn của việc sử dụng, quản lý, lưu trữ và duy trì các công cụ và các

tài sản test (ví dụ, quyết định về quy ước đặt tên cho các tập tin và các bài test, tạo các thư viện và xác định mô đun của phòng test)

- Assess whether the benefits will be achieved at reasonable cost - Đánh giá xem những lợi ích sẽ đạt được với chi phí hợp lý
    - Understanding the metrics that you wish the tool to collect and report, and configuring the tool to ensure these metrics can be captured and reported

6.2.3 Success Factors for Tools

- Rolling out the tool to the rest of the organization incrementally - Mở rộng các công cụ để phần còn lại của tổ chức từng bước
- Adapting and improving processes to fit with the use of the tool - Thích ứng và cải tiến quy trình để phù hợp với việc sử dụng các công cụ
- Providing training and coaching/mentoring for new users - Cung cấp đào tạo và huấn luyện / tư vấn cho người dùng mới
- Defining usage guidelines - Xác định các hướng dẫn sử dụng
- Implementing a way to gather usage information from the actual use - Triển khai thực hiện để thu thập thông tin sử dụng từ việc sử dụng thực tế
- Monitoring tool use and benefits - Giám sát việc sử dụng công cụ và lợi ích
- Providing support for the test team for a given tool - Cung cấp hỗ trợ cho các nhóm test cho một công cụ cho
- Gathering lessons learned from all teams - Bài học Gathering học được từ tất cả các đội