### **Giới thiệu về Testing, Objective &; Need for testing**

### 

### **Kiểm thử phần mềm là gì?**



#### Kiểm thử là quá trình được sử dụng để giúp xác định tính chính xác, đầy đủ và chất lượng của phần mềm máy tính.

#### Nói cách khác, kiểm thử là thực thi một hệ thống hoặc ứng dụng để tìm lỗi, lỗi hoặc lỗi phần mềm.

#### Mục đích là để kiểm tra xem phần mềm có đáp ứng các yêu cầu, nhu cầu và mong đợi cụ thể của khách hàng hoặc khách hàng hay không.

#### Ví dụ:

#### Các nhà sản xuất ô tô kiểm tra những chiếc xe về tốc độ tối đa, hiệu quả nhiên liệu và an toàn khỏi vụ tai nạn.

#### Những thử nghiệm này sau đó trở thành một phần của chiến lược quảng cáo cho doanh số bán xe.

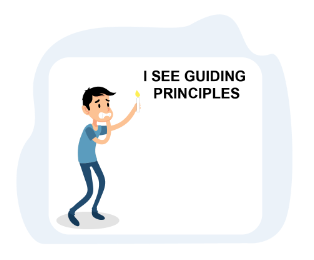
### **Mục tiêu &; Nhu cầu thử nghiệm**

  
Để phát hiện lỗi

* Phù hợp với sản phẩm theo đặc điểm kỹ thuật yêu cầu phần mềm (SRS). tức là chất lượng và bảo mật sản phẩm
* Để chịu chi phí và nỗ lực tối thiểu
* Thiết lập niềm tin vào phần mềm
* Đánh giá các thuộc tính của phần mềm cho các yêu cầu chức năng &; phi chức năng

### **Hãy đi sâu vào các nguyên tắc của Testing**

### **Nguyên tắc kiểm tra**

  
Kiểm thử phần mềm đặc biệt giàu trí tưởng tượng và là một nhiệm vụ trí tuệ để người kiểm thử thực hiện.

#### Kiểm thử phần mềm hoặc ứng dụng bao gồm một số nguyên tắc đóng vai trò quan trọng trong kiểm thử phần mềm trong khi kiểm thử dự án.

#### Hãy xem bảy nguyên tắc này từng nguyên tắc một,

* Thử nghiệm cho thấy sự hiện diện của các khuyết tật

#### Việc kiểm thử phần mềm nhằm mục đích làm cho chương trình bị sập. Giám sát máy tính làm giảm khả năng xảy ra lỗi.

* Thử nghiệm sớm là không thể

#### Đây là phương pháp kiểm tra chức năng của phần mềm trong tất cả các đầu vào và điều kiện tiên quyết có thể (hợp lệ hoặc không hợp lệ).

* Thử nghiệm sớm

#### Để tìm ra lỗi trong phần mềm, hoạt động kiểm thử sớm sẽ được bắt đầu.

### **Tiếp tục các nguyên tắc**

* Phân cụm khiếm khuyết

#### Ngay cả các mô-đun nhỏ cũng có thể chứa hầu hết các lỗi trong một dự án.

* Nghịch lý thuốc trừ sâu

#### Lặp đi lặp lại cùng một trường hợp kiểm thử sẽ không tiết lộ các lỗi mới. Tất cả các trường hợp kiểm thử cần được kiểm tra và các trường hợp kiểm thử được chèn hoặc thay đổi để xác định các lỗi mới.

* Thử nghiệm phụ thuộc vào ngữ cảnh

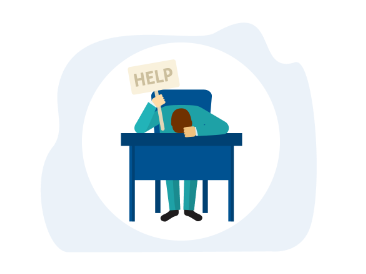
#### Cách tiếp cận thử nghiệm dựa trên bối cảnh của các ứng dụng được phát triển.

* Không có ngụy biện lỗi

#### Nếu một phần mềm được phát triển đáp ứng 99% nhưng không tuân theo yêu cầu của khách hàng. Chúng ta cần phải đáp ứng cả hai yêu cầu cùng một lúc.

### **Mô tả khiếm khuyết**

### **Khái niệm cơ bản về mô tả lỗi**

  
  
Một khiếm khuyết có thể là một biến thể hoặc sai lệch so với các yêu cầu kinh doanh ban đầu.

#### **Lỗi sau đó là gì?**

#### Trong khi đó, lỗi là hậu quả / kết quả của lỗi mã hóa.

#### Hai thuật ngữ này có một đường khác biệt rất mong manh, trong ngành cả hai đều là lỗi cần được sửa chữa và sau đó được sử dụng thay thế cho nhau bởi một số nhóm Kiểm thử.

#### Khi một tester thực hiện các trường hợp kiểm thử, anh ta có thể vấp phải kết quả kiểm thử mâu thuẫn với kết quả mong đợi.

#### Biến thể này trong kết quả kiểm tra được gọi là Lỗi phần mềm

**Vai trò của người kiểm thử phần mềm**

### **Vai trò của người kiểm thử phần mềm và các công ty liên quan**

  
Đã đến lúc giúp bạn có động lực, Làm thế nào?

#### Hãy để tôi cung cấp cho bạn một số chỉ định phổ biến nhất được cung cấp bởi các công ty khác nhau

1. Nhà phân tích QA (Tươi hơn)
2. Sr. QA Analyst (2-3 năm kinh nghiệm)
3. Điều phối viên nhóm QA (5-6 năm kinh nghiệm)
4. Test Manager (8-11 năm kinh nghiệm)
5. Quản lý kiểm thử cấp cao (14+ kinh nghiệm)

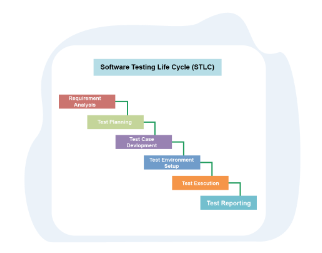
### **Các công ty lớn**

#### Có rất nhiều công ty lớn cung cấp dịch vụ kiểm thử cùng với các dịch vụ phát triển phần mềm cốt lõi khác. Họ đang:

1. ScienceSoft
2. Capgemini
3. Wipro
4. Cognizant
5. HP
6. Infosys
7. TCS
8. Hexaware

### **Mô hình vòng đời, tổng quan về STLC**

### **STLC là gì?**

  
  
Bạn đã đi xa đến mức này, hãy chờ đợi phần cuối của phần này.

#### VÒNG ĐỜI KIỂM THỬ PHẦN MỀM (STLC) là một chuỗi các hoạt động cụ thể được thực hiện trong quá trình kiểm thử để đảm bảo các mục tiêu chất lượng phần mềm được đáp ứng.

### **Các giai đoạn của STLC:**

* Phân tích yêu cầu
* Lập kế hoạch kiểm tra
* Phát triển trường hợp kiểm thử
* Thiết lập môi trường thử nghiệm
* Thực hiện kiểm thử
* Đóng chu kỳ thử nghiệm  
    
  Mỗi giai đoạn này có một tiêu chí Vào và Ra nhất định.

### **Tiêu chí nhập cảnh &; xuất cảnh**

#### Tiêu chí đầu vào: **Tiêu chí đầu vào** cung cấp các mục tiên quyết phải được hoàn thành trước khi thử nghiệm có thể bắt đầu.

#### Tiêu chí **thoát: Tiêu chí** thoát xác định các mục phải được hoàn thành trước khi thử nghiệm có thể được kết thúc

#### Chúng tôi sẽ thảo luận thêm về các chủ đề này trong khóa học của chúng tôi.

#### Tôi hy vọng bạn đã học được điều gì đó trong chủ đề này, đến cuối loạt bài này, bạn sẽ sẵn sàng kiểm tra bản thân trong thế giới thử nghiệm.

### **Kiểm tra thủ công**

### **Giới thiệu về kiểm thử thủ công**



#### Kiểm thử thủ công là một loại kiểm thử phần mềm mà người kiểm thử tiến hành thủ công mà không cần sử dụng các công cụ tự động

#### Kiểm thử thủ công là một quá trình tìm ra các khiếm khuyết hoặc lỗi trong một chương trình phần mềm hoặc hệ thống.

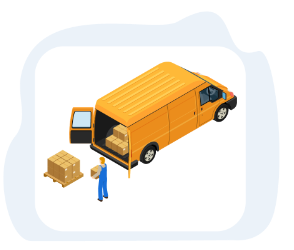
#### Người kiểm tra đóng vai trò người dùng cuối quan trọng và xác minh rằng tất cả các tính năng của ứng dụng đang chạy đúng.

### **Ưu điểm của kiểm tra thủ công**



1. Nhận phản hồi trực quan nhanh chóng và chính xác
2. Nó rẻ hơn vì bạn không phải chi ngân sách cho các công cụ và quy trình tự động hóa
3. Phán đoán và trực giác của con người luôn có lợi cho yếu tố thủ công

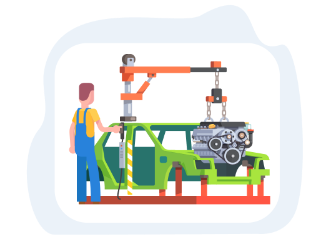
### **Nhược điểm của kiểm tra thủ công**



1. Quá trình kiểm tra thủ công không thể được đăng ký, do đó không thể sử dụng lại thử nghiệm thủ công.
2. Nhiều tác vụ khó thực hiện thủ công và có thể mất thêm thời gian bước kiểm thử phần mềm.
3. Một hình thức nghiên cứu kém hiệu quả hơn, như một con người, thực hiện nó. Nó cũng dễ bị tổn thương bởi những sai lầm và khiếm khuyết.

### **Kiểm thử tự động hóa**

### **Giới thiệu**

  
Chúng ta đã tìm hiểu về kiểm thử thủ công, hãy xem tự động hóa có thể làm gì.

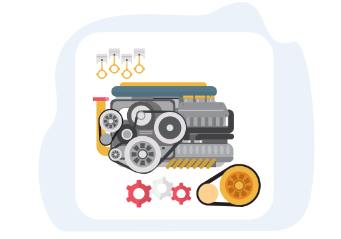
#### Kiểm thử tự động hóa có nghĩa là tiến hành bộ trường hợp kiểm thử bằng cách sử dụng một công cụ tự động hóa.

#### Trong phương pháp này, các kịch bản kiểm thử được thực thi bởi người kiểm tra và tự động tạo ra kết quả kiểm tra bằng phần mềm tự động hóa.

#### Kiểm thử tự động cho phép bạn chạy các bài kiểm tra hồi quy và hoạt động lặp đi lặp lại mà không cần sự can thiệp của người kiểm tra thủ công.

#### Kiểm thử tự động chỉ dựa vào thử nghiệm được viết sẵn chạy tự động để khớp các thử nghiệm thực với kết quả dự đoán.

### **Ưu điểm của kiểm thử tự động hóa**

  
 Nó có thể dễ dàng tăng năng suất, vì nó mang lại kết quả kiểm tra nhanh chóng và chính xác

* Kiểm thử tự động có thể giúp bạn tìm thấy nhiều lỗi hơn người kiểm tra
* Quá trình tự động hóa có thể được ghi lại. Điều này cho phép bạn tái sử dụng và
* thực hiện cùng một loại hoạt động thử nghiệm

### **Nhược điểm của kiểm thử tự động hóa**

  
 Các công cụ để chạy kiểm thử tự động hóa có thể tốn kém, điều này sẽ làm tăng chi phí của dự án nghiên cứu.

* Gỡ lỗi tập lệnh kiểm thử là một vấn đề lớn khác trong kiểm thử tự động. Bảo trì thử
* nghiệm rất tốn kém.
* Các công cụ tự động hóa có những hạn chế làm giảm phạm vi tự động hóa.

### **Sự khác biệt giữa Tự động hóa &; Kiểm thử thủ công**

### **Sự khác biệt lớn**

* Quá trình kiểm tra thủ công không chính xác vì khả năng lỗi của con người trong khi quy trình
* Tự động hóa là đáng tin cậy vì nó dựa trên mã và tập lệnh.
* Kiểm thử thủ công là một quá trình cần có thời gian, trong khi kiểm thử tự động hóa rất dễ dàng.  
    
  Kiểm thử thủ công là có thể mà không cần kiến thức về lập trình trong khi kiểm thử tự động hóa là không thể nếu không có kiến thức về lập trình.
* Kiểm thử thủ công cho phép Kiểm thử ngẫu nhiên trong khi Kiểm thử tự động hóa không cho phép Kiểm thử ngẫu nhiên.

### **Kiểm thử nhanh**

### **Thông tin chi tiết cơ bản**

#### Đây là một phương pháp phát triển lặp đi lặp lại và các yêu cầu phát triển thông qua sự hợp tác giữa khách hàng và các nhóm tự tổ chức.

#### 

#### Agile Testing có thể bắt đầu khi bắt đầu dự án với sự tích hợp liên tục giữa phát triển và thử nghiệm.

#### 

#### Agile Testing có khung thời gian ngắn hơn, được gọi là lặp lại.

* **Ưu điểm của agile testing**

1. Kiểm thử nhanh giúp tiết kiệm thời gian và tiền bạc.
2. Ít tài liệu hơn, phản hồi thường xuyên từ người dùng cuối.
3. Các cuộc họp hàng ngày có thể giúp xác định các vấn đề trước.

* **Nguyên tắc kiểm thử Agile**
* 

#### Các nguyên tắc kiểm thử khác nhau bao gồm:

* Thử nghiệm liên tục:

#### Nhóm Agile tiến hành kiểm thử liên tục vì đây là cách duy nhất để đảm bảo sản phẩm tiếp tục phát triển.

* Phản hồi liên tục:

#### Thử nghiệm nhanh cung cấp phản hồi liên tục để sản phẩm của bạn có thể đáp ứng nhu cầu kinh doanh.

* Các thử nghiệm được thực hiện bởi cả nhóm:

#### Các nhà phát triển, nhóm phân tích và thử nghiệm kinh doanh đều kiểm tra ứng dụng.

### **Tiếp tục các nguyên tắc của Agile testing**

* Giảm thời gian phản hồi phản hồi:

#### Trong kiểm thử nhanh, nhóm kinh doanh tham gia vào mọi lần lặp. Do đó, phản hồi liên tục làm giảm thời gian phản hồi.

* Mã đơn giản & sạch:

#### Trong cùng một lần lặp, các khiếm khuyết do nhóm Agile đặt ra được sửa chữa. Điều này giúp giữ cho mã an toàn và đơn giản hóa.

* Ít tài liệu hơn:

#### Danh sách kiểm tra có thể tái sử dụng và nhóm tập trung vào thử nghiệm

* Lái thử:

#### Trong thời gian thực hiện, người ta phải chạy thử nghiệm.

### **Chiến lược & kế hoạch kiểm thử**

### **Chiến lược kiểm thử giới thiệu**

  
Tất cả chúng ta đều biết rất rõ rằng không có gì hoạt động trừ khi chúng ta có một chiến lược phù hợp.

#### Chúng tôi có chiến lược trong Thể thao, Kinh doanh, Chiến tranh, v.v

### **Chiến lược thử nghiệm**

#### Vậy Chiến lược kiểm thử là gì?

#### Chiến lược kiểm thử là một tài liệu cấp cao nắm bắt cách tiếp cận về cách chúng tôi thử nghiệm sản phẩm và đạt được các mục tiêu.

#### Nó được phát triển bởi người quản lý dự án.

#### Nó là một tài liệu tĩnh. Một khi được xác định, nó không thể thay đổi.

#### Nó được xác định về mặt tổ chức và có thể được sử dụng cho các dự án tương tự khác.

### **Kế hoạch kiểm tra giới thiệu**

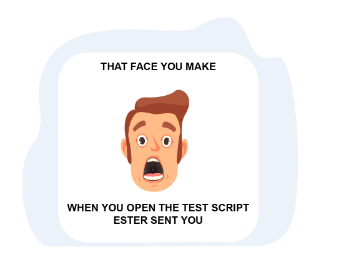
#### Tài liệu kế hoạch thử nghiệm là một tài liệu chứa kế hoạch cho tất cả các hoạt động thử nghiệm sẽ được thực hiện để cung cấp một sản phẩm chất lượng.

#### Nó được chuẩn bị bởi trưởng nhóm kiểm tra hoặc người quản lý thử nghiệm.

#### Nó là một tài liệu động. Nó có thể được thay đổi.

#### Ở cấp độ dự án, nó được thành lập.

### **Các trường hợp kiểm thử / tập lệnh**

  
Các trường hợp kiểm thử là tập hợp các bước thực thi tích cực và tiêu cực của một kịch bản kiểm thử có tập hợp các điều kiện tiên quyết, dữ liệu thử nghiệm, kết quả mong đợi, hậu điều kiện và kết quả thực tế.

#### Trong một ngôn ngữ giáo dân, chúng ta có thể nói

#### Câu trả lời của Test Case "How to be tested"

#### Mục đích của trường hợp kiểm thử là xác nhận kịch bản kiểm thử bằng cách thực hiện một tập hợp các bước.

### **Mẫu cơ bản của test case**

  
Một trường hợp kiểm thử có thể bao gồm các yếu tố dưới đây.

* Test Case ID: ID của test case.
* Test Case Summary: Tóm tắt / mục tiêu của test case.
* Yêu cầu liên quan: ID của yêu cầu mà trường hợp kiểm thử này liên quan / theo dõi đến.
* Điều kiện tiên quyết: Bất kỳ điều kiện tiên quyết hoặc điều kiện tiên quyết nào phải được đáp ứng trước
* khi thực hiện thử nghiệm.
* Quy trình kiểm tra: Quy trình từng bước để thực hiện kiểm thử.

### **Tiếp tục mẫu cơ bản của test case**

* Kết quả mong đợi: Kết quả dự kiến của bài kiểm tra
* Kết quả thực tế: Kết quả thực tế của bài kiểm tra; sẽ được điền sau khi thực hiện bài kiểm tra.
* Tình trạng: Đạt hoặc không đạt. Các trạng thái khác có thể là 'Không được thực thi' nếu thử nghiệm
* không được thực hiện và 'Bị chặn' nếu thử nghiệm bị chặn.
* Ghi chú: Bất kỳ nhận xét nào về trường hợp kiểm thử hoặc thực hiện kiểm thử.
* Ngày thực hiện: Ngày thực hiện bài kiểm tra.
* Môi trường kiểm thử: Môi trường (Phần cứng / Phần mềm / Mạng) trong đó kiểm thử được thực hiện.
* Dữ liệu thử nghiệm: Dữ liệu thử nghiệm hoặc liên kết đến dữ liệu thử nghiệm sẽ được sử dụng trong khi tiến hành thử nghiệm.

### **Ví dụ**

  
Hãy thử hiểu với ví dụ thời gian thực:

#### Giả sử rằng chúng ta cần kiểm tra chức năng của trang đăng nhập của ứng dụng Gmail. Các trường hợp kiểm thử cho chức năng trang đăng nhập ở trên như sau:

#### Ví dụ về trường hợp kiểm thử:

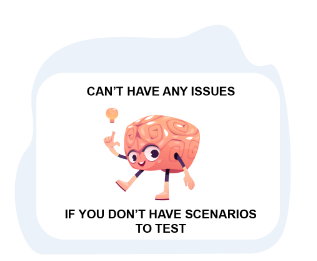
#### Test Case 1: Nhập User Name hợp lệ và Password hợp lệ

#### Test Case 2: Nhập User Name hợp lệ và Password không hợp lệ

#### Test Case 3: Nhập Tên người dùng không hợp lệ và Mật khẩu hợp lệ

#### Test Case 4: Nhập tên người dùng không hợp lệ và mật khẩu không hợp lệ

### **Kịch bản thử nghiệm**

  
Kiểm tra có nghĩa là xác minh hoặc xác thực và Kịch bản có nghĩa là bất kỳ hành trình nào với người dùng. Nó cho biết xác minh hành trình của người dùng khi kết hợp.

#### Thường được đặt tên là Điều kiện thử nghiệm hoặc Xác suất thử nghiệm.

#### Test Scenario đưa ra ý tưởng về những gì chúng ta phải kiểm tra. Test Scenario giống như một test case cấp cao.

#### Câu trả lời tình huống kiểm tra "**Những gì cần được kiểm tra**"

#### Kiểm tra có nghĩa là xác minh hoặc xác thực và Kịch bản có nghĩa là bất kỳ hành trình nào với người dùng. Nó cho biết xác minh hành trình của người dùng khi kết hợp.

### **Tại sao chúng ta cần nó?**

1. Tạo các kịch bản thử nghiệm đảm bảo phạm vi thử nghiệm đầy đủ.
2. Kịch bản kiểm thử là rất quan trọng để nghiên cứu chức năng end-to-end của chương trình.
3. Chúng phục vụ như một công cụ nhanh chóng để xác định nỗ lực làm việc thử nghiệm và do đó thiết lập một đề xuất cho khách hàng.
4. Chúng giúp đánh giá các giao dịch đầu cuối quan trọng.

### **Ví dụ**

#### Hãy lấy một ví dụ:

#### Giả sử rằng chúng ta cần kiểm tra chức năng của trang đăng nhập của ứng dụng Gmail.

#### Kịch bản thử nghiệm cho chức năng trang đăng nhập Gmail như sau:

#### Kịch bản kiểm tra: Xác thực trang đăng nhập

* Test Case 1: Nhập tên người dùng và mật khẩu hợp lệ
* Test Case 2: Đặt lại mật khẩu của bạn
* Test Case 3: Nhập thông tin đăng nhập không hợp lệ

### **Đặc tả trường hợp kiểm thử**

  
Tài liệu Test Case Specification mô tả một bản tóm tắt chi tiết về:

#### Những kịch bản nào sẽ được kiểm tra

#### Chúng sẽ được kiểm tra như thế nào

#### Bao lâu họ sẽ được kiểm tra, vân vân và vân vân

#### Kế hoạch kiểm thử là tập hợp tất cả các Thông số kỹ thuật kiểm tra

### **Cần gì?**

#### Cách xác định test case theo cách duy nhất như sau:

* Mục tiêu trường hợp kiểm thử: Mục đích của kiểm thử
* Mục kiểm tra: Các mục (ví dụ: thông số kỹ thuật yêu cầu, thông số kỹ thuật thiết kế, mã, v.v.) cần thiết để chạy một trường hợp kiểm thử cụ thể. Điều này nên được cung cấp trong tính năng "Ghi chú" hoặc
* "Tệp đính kèm". Nó mô tả các tính năng và điều kiện cần thiết để thử nghiệm.

### **Tiếp tục nhu cầu đặc tả trường hợp**

* Thông số kỹ thuật đầu vào: Mô tả những gì được yêu cầu (từng bước) để thực hiện trường hợp kiểm thử (ví dụ: tệp đầu vào, giá trị phải được nhập vào trường, v.v.). Điều này nên được cung cấp trong trường "Hành động".
* Thông số kỹ thuật đầu ra: Mô tả hệ thống sẽ trông như thế nào sau khi trường hợp kiểm thử được chạy.
* Điều này nên được cung cấp trong trường "Kết quả mong đợi".

**Ma trận truy xuất nguồn gốc yêu cầu (RTM)**



#### **Ma trận truy xuất nguồn gốc** yêu cầu (RTM) là một tài liệu ánh xạ và theo dõi các yêu cầu của người dùng với các trường hợp kiểm thử.

#### Nó chứa tất cả các yêu cầu theo kế hoạch của khách hàng và tiêu chí truy xuất nguồn gốc trong một văn bản duy nhất, được phân phối vào cuối vòng đời của Phát triển phần mềm.

#### Mục đích chính của RTM là xác minh rằng tất cả các thông số kỹ thuật được kiểm tra thông qua các trường hợp kiểm thử để không có tính năng nào bị bỏ chọn trong quá trình Kiểm thử phần mềm.

### **Các loại ma trận kiểm tra truy xuất nguồn gốc**

#### Có ba loại ma trận truy xuất nguồn gốc:

* Truy xuất nguồn gốc chuyển tiếp:

#### Điều này là cần thiết để kiểm tra xem dự án có tiến triển đúng hướng và đúng sản phẩm hay không.

* Truy xuất nguồn gốc ngược hoặc ngược:

#### Mục đích của truy xuất nguồn gốc ngược là để xác minh rằng chúng tôi không mở rộng phạm vi của dự án bằng cách thêm mã, các yếu tố thiết kế, thử nghiệm hoặc công việc khác không được quy định trong các yêu cầu

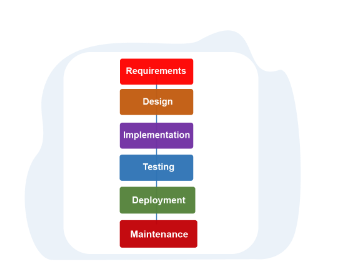
* Truy xuất nguồn gốc hai chiều (Tiến &; lùi):

#### Điều này đảm bảo yêu cầu được bao phủ bởi các trường hợp kiểm thử. Nó phân tích ảnh hưởng của sự thay đổi thông số kỹ thuật của sản phẩm làm việc bị ảnh hưởng bởi lỗi và ngược lại.

### **Ưu điểm của RTM**

1. 100% phạm vi kiểm tra.
2. Nó cho phép dễ dàng xác định các chức năng bị thiếu.
3. Nó cho phép xác định các trường hợp kiểm thử cần được cập nhật trong trường hợp thay đổi yêu cầu.
4. Trạng thái thực hiện tổng thể của các thử nghiệm rất dễ theo dõi

### **Mô hình thác nước**

  
Tôi hy vọng bạn đã vui vẻ cho đến bây giờ, ở cuối chương này, bạn sẽ là 50% Tester.

#### Mô hình thác nước được thực hiện như mô hình hệ thống đầu tiên.

#### Mô hình thác nước là một mô hình tuần tự chia phát triển phần mềm thành các giai đoạn được xác định trước

#### Trong mô hình thác nước, mỗi bước phải được hoàn thành trước khi bước tiếp theo có thể bắt đầu để các giai đoạn không chồng chéo với nhau.

**Các giai đoạn của mô hình thác nước**

#### Các pha tuần tự trong mô hình Thác nước là:

* Phân tích yêu cầu:

#### Yêu cầu chi tiết của hệ thống phần mềm sẽ được phát triển & thu thập từ khách hàng.

* Thiết kế hệ thống:

#### Các yêu cầu được nghiên cứu từ giai đoạn trước và nó giúp xác định kiến trúc hệ thống.

* Thực hiện:

#### Hệ thống được thiết kế ban đầu trong các chương trình nhỏ gọi là đơn vị. Mỗi đơn vị được phát triển và thử nghiệm theo chức năng của nó.

**Các giai đoạn của mô hình thác nước**

* Kiểm thử hệ thống:

#### Trong bước này, bạn đang kiểm tra chương trình để xác minh rằng nó được xây dựng theo yêu cầu của khách hàng.

* Triển khai hệ thống:

#### Sau khi kiểm thử chức năng và phi chức năng được thực hiện; Sản phẩm được triển khai hoặc phát hành ra thị trường trong môi trường khách hàng.

* Bảo trì:

#### Khi chương trình của bạn đã sẵn sàng để sử dụng, bạn có thể cần phải thay đổi mã sau này khi cần theo yêu cầu của khách hàng.

### **Mô hình lặp**

  
  
Trong mô hình này, sản phẩm được thiết kế, triển khai và thử nghiệm từng bước.

#### Ít hơn được thêm vào mỗi lần trước khi sản phẩm được hoàn thành.

#### Sản phẩm được phân hủy thành nhiều bộ phận khác nhau, mỗi bộ phận được thiết kế và chế tạo riêng.

#### Nhiều chu kỳ phát triển diễn ra ở đây, làm cho vòng đời trở thành một chu kỳ đa thác nước.

### **Ưu điểm của mô hình lặp**

1. Mô hình gia tăng cho phép sử dụng một phần sản phẩm và tránh thời gian phát triển dài.
2. Tạo phần mềm làm việc nhanh chóng và sớm trong suốt vòng đời phần mềm.
3. Mô hình này linh hoạt hơn và ít tốn kém hơn để thay đổi phạm vi và yêu cầu.
4. Việc kiểm tra và gỡ lỗi sẽ dễ dàng hơn vì những thay đổi nhỏ hơn được thực hiện trong mỗi lần lặp.

### 

### **Nhược điểm của mô hình lặp**

1. Nó cần lập kế hoạch và thiết kế tốt ở mọi bước.
2. Cần một định nghĩa rõ ràng và đầy đủ về toàn bộ hệ thống trước khi nó có thể được chia nhỏ và xây dựng dần dần.
3. Tổng chi phí cao hơn thác nước.

### **Mô hình xoắn ốc**

  
Mô hình xoắn ốc là sự kết hợp giữa mô hình thác nước và mô hình lặp.

#### Mỗi quá trình mô hình xoắn ốc bắt đầu với một mục tiêu thiết kế và kết thúc với việc khách hàng xem xét tiến độ.

#### Mô hình này được sử dụng tốt nhất cho các dự án lớn cần cải tiến liên tục.

#### Các hoạt động tương tự được lặp lại cho tất cả các xoắn ốc cho đến khi tất cả các phần mềm được xây dựng.

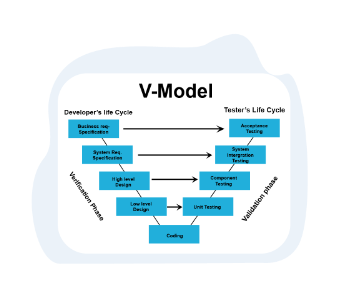
### **Ưu điểm của mô hình xoắn ốc**

1. Ước tính chi phí trở nên dễ dàng vì việc xây dựng nguyên mẫu được thực hiện trong các phân đoạn nhỏ.
2. Phát triển nhanh chóng và các tính năng được thêm vào một cách có hệ thống.
3. Luôn có chỗ cho phản hồi của khách hàng và các thay đổi được thực hiện nhanh hơn.

### **Nhược điểm của mô hình xoắn ốc**

1. Nó chỉ hoạt động tốt nhất cho các dự án lớn và có thể tốn kém cho các dự án nhỏ.
2. Phân tích rủi ro là một giai đoạn quan trọng vì vậy nó đòi hỏi những người chuyên gia để đánh giá rủi ro.
3. Tài liệu là nhiều hơn như vậy vì nó có một số lượng lớn các giai đoạn trung gian.

### **V- Mô hình**

  
V-Model là mô hình SDLC có giai đoạn thử nghiệm tương ứng với mọi giai đoạn phát triển trong mô hình thác nước.

#### Nó còn được gọi là Mô hình xác minh và xác nhận.

#### Sự phát triển của nó được xoa song song. Vì vậy, có các giai đoạn Xác minh ở một bên của giai đoạn 'V' và Xác thực ở phía bên kia.

#### Mô hình V là một phần mở rộng của mô hình thác nước và dựa trên sự liên kết cho từng giai đoạn phát triển tương ứng của giai đoạn thử nghiệm.

### **Ưu điểm của mô hình V**

1. Đây là một mô hình có tính kỷ luật cao và các giai đoạn được hoàn thành từng giai đoạn một.
2. Đơn giản, dễ hiểu và dễ sử dụng.
3. Hoạt động tốt cho các dự án nhỏ, nơi các yêu cầu dễ hiểu.

**Nhược điểm của mô hình V**

1. Rất cứng nhắc và kém linh hoạt nên việc điều chỉnh phạm vi rất khó khăn và tốn kém.
2. Phần mềm được phát triển trong giai đoạn thực hiện, vì vậy không có nguyên mẫu ban đầu của phần mềm được sản xuất.
3. Không phù hợp với các dự án có yêu cầu có nguy cơ thay đổi từ trung bình đến cao.

### **Mô hình nguyên mẫu**

  
Một nguyên mẫu được thiết kế để thử nghiệm và thử một thiết kế mới cho các nhà phát triển hệ thống và người dùng để cải thiện độ chính xác.

#### Một nguyên mẫu được chế tạo, thử nghiệm và sau đó làm lại khi cần thiết cho đến khi đạt được một nguyên mẫu chấp nhận được.

#### Các mô hình tạo mẫu phần mềm hoạt động tốt nhất trong các tình huống mà các yêu cầu của dự án không được biết đến.

#### Đây là một phương pháp lặp, thử và lỗi diễn ra giữa nhà phát triển và máy khách.

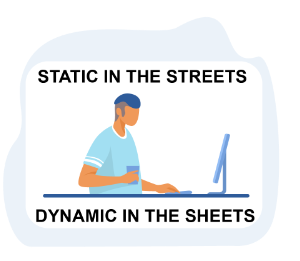
### **Ưu điểm của mô hình Nguyên mẫu**

1. Đôi khi nó giúp chứng minh khái niệm cho các nhà đầu tư tiềm năng để có được tài trợ cho một dự án.
2. Nó làm giảm nguy cơ thất bại, vì rủi ro tiềm ẩn có thể được xác định sớm và các bước khắc phục có thể được thực hiện.
3. Sự lặp lại giữa nhóm phát triển và khách hàng cung cấp một môi trường rất tốt và dẫn điện trong suốt dự án.

### **Nhược điểm của mô hình nguyên mẫu**

1. Nếu quá nhiều thay đổi được yêu cầu trong sản phẩm mẫu hoặc mô hình, nó có thể làm xáo trộn nhịp điệu của nhóm phát triển.
2. Phương pháp này có thể làm tăng độ phức tạp của hệ thống vì phạm vi của hệ thống có thể mở rộng vượt ra ngoài kế hoạch ban đầu.

### **Kiểm tra động**

  
Tất cả chúng ta đều biết về anh chàng này, tôi không muốn khoe khoang về điều này nhưng phần mềm có thể kiếm được hàng tỷ đô la. Ngay cả nó đã làm cho tôi.

#### Hãy thử và lặp lại lịch sử này.

#### Thử nghiệm động?

#### Nó còn được gọi là Validation in Software Testing.

#### Phần mềm Dynamic Testing được thực thi bằng cách sử dụng một tập hợp các giá trị đầu vào và đầu ra của nó sau đó được kiểm tra và so sánh với những gì được mong đợi.

#### Kiểm tra, đánh giá, hướng dẫn có liên quan đến thử nghiệm này.

#### Mã phải được thực thi để kiểm tra hành vi động của phần mềm.

### **Các loại kiểm thử động**

#### Dynamic Testing được phân thành ba loại:

1. Kiểm tra hộp trắng
2. Thử nghiệm hộp đen
3. Kiểm tra hộp xám

#### Chúng ta sẽ thảo luận kỹ lưỡng về cả hai loại này trong chương tới.

### **Ưu điểm của Dynamic Testing**

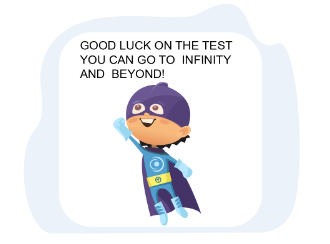
  
  
Bài tập cấp cao được coi là:

#### Nó luôn có thể tìm thấy các lỗi mà kiểm tra tĩnh không thể tìm thấy và đó là lý do tại sao nó được coi là bài tập cấp cao.

* Cải thiện chất lượng:

#### Thực thi phần mềm dẫn đến cơ hội tìm thấy nhiều lỗi hơn trong ứng dụng, vì vậy nó đảm bảo phần mềm không có lỗi ở một mức độ nào đó.

### **Nhược điểm của kiểm tra động**

  
Tốn thời gian:

#### Đây là một nhiệm vụ tốn thời gian vì mục đích của nó là thực thi phần mềm và kết quả là, cần nhiều trường hợp kiểm thử hơn để thực thi.

* Tăng giá thành sản phẩm:

#### Nó không được thực hiện sớm trong vòng đời phần mềm và do đó các lỗi được sửa trong các giai đoạn sau dẫn đến nhiều chi phí hơn.

* Thực hiện tốn kém:

#### Nó đòi hỏi nhiều nhân lực hơn để hoàn thành nhiệm vụ.

### **Kiểm tra tĩnh**

#### Kiểm thử tĩnh còn được gọi là Xác minh trong Kiểm thử phần mềm.

#### Kiểm thử tĩnh là một kỹ thuật kiểm thử phần mềm mà chúng ta có thể kiểm tra các lỗi trong phần mềm mà không thực sự thực hiện nó.

#### Kiểm thử tĩnh được thực hiện để ngăn ngừa các khiếm khuyết phát triển sớm, vì việc xác định nguồn gốc của các lỗi dễ dàng hơn so với chính các thất bại.

#### Kiểm thử tĩnh giúp tìm ra các lỗi mà Dynamic Testing không thể tìm thấy.

### **Ưu điểm của kiểm thử tĩnh**

1. Vì thử nghiệm tĩnh có thể bắt đầu sớm trong vòng đời, phản hồi sớm về các vấn đề chất lượng có thể được thiết lập.
2. Bằng cách phát hiện lỗi ở giai đoạn đầu, chi phí làm lại thường tương đối thấp.
3. Các thử nghiệm tĩnh góp phần nâng cao nhận thức về các vấn đề chất lượng.

### **Nhược điểm của kiểm thử tĩnh**

1. Tốn thời gian như được thực hiện thủ công
2. Chuyên gia có tay nghề hạn chế để tiến hành phân tích mã kỹ lưỡng.

### **Kiểm tra chức năng**

### **Kiểm tra chức năng**



#### Kiểm thử chức năng liên quan đến việc kiểm thử ứng dụng theo yêu cầu kinh doanh.

#### Mục đích của kiểm thử chức năng là để xác minh rằng ứng dụng hoạt động theo cách nó được dự định.

#### Kiểm tra chức năng đảm bảo chương trình đáp ứng đúng các thông số kỹ thuật.

### **Các loại kiểm thử chức năng**

  
**Kiểm thử đơn vị**: Kiểm thử riêng lẻ và độc lập các phần nhỏ nhất có thể kiểm tra của một ứng dụng.

#### Kiểm thử tích hợp: Khi các mô-đun phần mềm riêng lẻ được kết hợp với nhau và kiểm thử thành một nhóm thì nó được gọi là **Kiểm thử tích hợp**

#### **Kiểm thử khói**: Thử nghiệm sơ bộ để tiết lộ các lỗi đơn giản đủ nghiêm trọng để (ví dụ) từ chối bản phát hành hoặc bản dựng phần mềm tiềm năng.

#### **Kiểm tra sự tỉnh táo**: Chạy qua các chức năng rất ngắn để đảm bảo rằng một phần của hệ thống hoặc phương pháp hoạt động gần như mong đợi.

#### **Kiểm thử** hệ thống: Thử nghiệm được thực hiện trên một hệ thống tích hợp, hoàn chỉnh để đánh giá sự tuân thủ của hệ thống với các yêu cầu cụ thể của nó.

#### **Kiểm thử hồi quy**: Kiểm tra lại hệ thống phần mềm để xác nhận rằng những thay đổi được thực hiện đối với một số phần của mã không có bất kỳ tác dụng phụ nào đối với các chức năng hệ thống hiện có.

### **Kiểm thử phi chức năng**

### **Giới thiệu**

#### Mỗi bên trái đều có bên phải của nó.

#### Thử nghiệm phi chức năng được thiết kế để tìm hiểu xem sản phẩm của bạn có cung cấp trải nghiệm người dùng tốt hay không.

#### Trọng tâm chính là cải thiện trải nghiệm người dùng về tốc độ hệ thống phản hồi yêu cầu.

#### Kiểm thử phi chức năng đề cập đến các khía cạnh khác nhau của phần mềm như hiệu suất, tải, căng thẳng, khả năng mở rộng, bảo mật, khả năng tương thích, v.v.,

### **Các loại phi chức năng**

  
**Kiểm thử** hiệu suất: Để đánh giá hiệu suất của các thành phần của một hệ thống cụ thể theo một khối lượng công việc cụ thể.

#### **Kiểm** tra tải: Kiểm tra hành vi của hệ thống dưới một tải cụ thể hoặc để có được điểm hòa vốn nơi hệ thống bắt đầu hạ cấp hiệu suất của nó.

#### **Kiểm tra căng thẳng**: Nó được thực hiện để tìm công suất giới hạn trên của hệ thống và cũng để xác định cách hệ thống hoạt động nếu tải hiện tại vượt quá mức tối đa dự kiến.

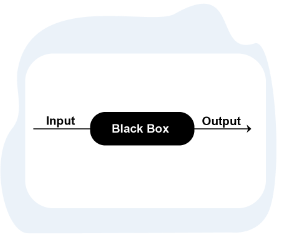
### **Các loại phi chức năng**

#### **Kiểm thử khả năng sử dụng**: Kiểm thử để xác định mức độ hiểu sản phẩm phần mềm, dễ học, dễ vận hành và hấp dẫn người dùng trong các điều kiện cụ thể.

#### **Kiểm tra bảo mật**: Điều này nhằm phát hiện ra các lỗ hổng của hệ thống và xác định rằng dữ liệu và tài nguyên của nó được bảo vệ khỏi những kẻ xâm nhập có thể.

#### **Kiểm tra tính di** động: Độ tin cậy của phần mềm là xác suất phần mềm sẽ hoạt động bình thường trong một môi trường cụ thể và trong một khoảng thời gian nhất định.

### **Thử nghiệm hộp đen**

  
BLACK BOX TESTING được mô tả như một kỹ thuật kiểm thử trong đó Ứng dụng

#### Chức năng Under Test (AUT) được kiểm tra mà không cần xem cấu trúc mã nội bộ, chi tiết triển khai và kiến thức về các lộ trình nội bộ của phần mềm.

#### Trong BlackBox Testing, chúng tôi chỉ tập trung vào đầu vào và đầu ra của hệ thống phần mềm mà không phải lo lắng về các chi tiết bên trong của chương trình phần mềm.

#### Nó còn được gọi là Kiểm tra hành vi / dựa trên đặc điểm kỹ thuật / đầu vào-đầu ra.

#### Mục đích chính của thử nghiệm Black Box là kiểm tra xem phần mềm có hoạt động như mong đợi và đáp ứng yêu cầu của khách hàng hay không.

### **Ví dụ**

  
Hãy hiểu điều này với một ví dụ:

#### Trong trường hợp của Google hoặc bất kỳ công cụ tìm kiếm nào khác, người dùng nhập văn bản vào trình duyệt.

#### Công cụ tìm kiếm định vị và truy xuất kết quả dữ liệu.

#### Người dùng không nhận thức được nguồn gốc của quá trình.

#### Người kiểm tra phần mềm hộp đen chọn một tập hợp các đầu vào hợp lệ và không hợp lệ và kiểm tra các phản hồi đầu ra hợp lệ.

### **Ưu điểm của kiểm thử hộp đen**



1. Nó rất phù hợp và hiệu quả cho các phân khúc quy mô lớn.
2. Một tester có thể là phi kỹ thuật. Người kiểm tra có trình độ cao không bắt buộc và do đó nó ít tốn kém hơn.
3. Các thử nghiệm được thực hiện theo quan điểm của người dùng và rất hữu ích để xác minh sự khác biệt trong thực tế và hệ thống và thông số kỹ thuật.

### **Nhược điểm của kiểm tra hộp đen**

1. Phạm vi bảo hiểm bị hạn chế vì chỉ có một số thử nghiệm được chọn thực sự được thực hiện.
2. Vì tester có kiến thức hạn chế về phần mềm, việc kiểm thử không hiệu quả.
3. Phạm vi bảo hiểm bị mù vì người kiểm tra không thể nhắm mục tiêu các mã cụ thể.

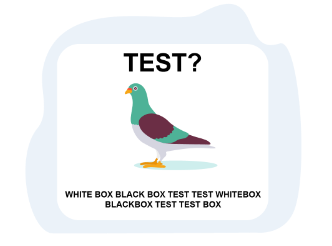
### **Kiểm tra hộp trắng**

  
Đây là phương pháp kiểm tra trong đó người kiểm tra nhận thức được các mã nội bộ và bố cục của chương trình.

#### Người kiểm tra chọn đầu vào và thực hiện kiểm tra bằng cách cung cấp cho đầu vào thiết bị thông qua các mã và xác định đầu ra chính xác.

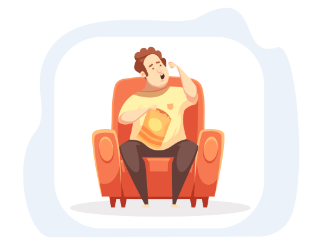
#### Trọng tâm chính của **WBT** là tăng cường bảo mật và cải thiện thiết kế và khả năng sử dụng của phần mềm.

### **Ví dụ**

  
Làm thế nào về một ví dụ:

#### Thử nghiệm hộp trắng giống như công việc của một thợ sửa xe, người biết tất cả các bên trong của chiếc xe và kiểm tra lý do tại sao nó không hoạt động hoặc cho số dặm ít hơn, v.v.

### **Ưu điểm của thử nghiệm hộp trắng**



1. Kiểm thử kỹ lưỡng hơn, vì nó bao gồm tất cả các đường dẫn mã có thể.
2. Người kiểm thử nhận thức được cấu trúc mã hóa nội bộ, sau đó sẽ rất hữu ích khi quyết định loại dữ liệu đầu vào nào là cần thiết để kiểm tra hiệu quả các ứng dụng phần mềm.
3. White Box Testing cho phép bạn giúp tối ưu hóa mã.

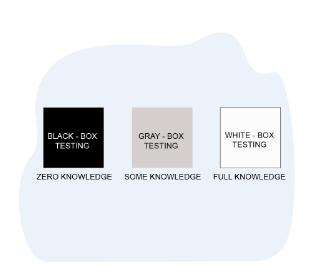
**Nhược điểm của kiểm tra hộp trắng**

****

### **Nhược điểm của kiểm tra hộp trắng**

1. **Vì cần có một nguồn lực có tay nghề cao để thực hiện kiểm thử, những người biết kiến thức sâu về cấu trúc nội bộ của mã, chi phí cao.**
2. **Nếu ứng dụng được thử nghiệm có kích thước lớn, thì việc kiểm tra toàn diện là không thể.**
3. **Không thể kiểm thử để kiểm tra từng đường dẫn / điều kiện của chương trình phần mềm, điều này có thể bỏ lỡ các lỗi trong mã.**

### **Thử nghiệm hộp màu xám**

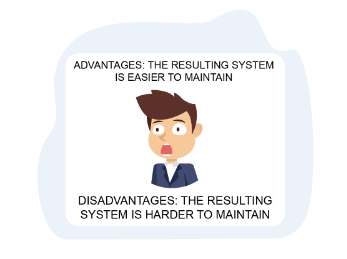
****Nó là sự kết hợp giữa thử nghiệm Black Box và thử nghiệm White Box.

#### Trong thử nghiệm Gray Box, tester có kiến thức về một số phần của cấu trúc bên trong.

#### Điều này đòi hỏi phải có quyền truy cập vào các cấu trúc dữ liệu nội bộ và thuật toán để xây dựng các trường hợp kiểm thử.

#### Việc kiểm tra Gray Box được coi là không xâm phạm và vô tư vì nó không cho phép người kiểm tra có quyền truy cập vào mã nội bộ.

### **Ưu điểm của kiểm tra hộp xám**

**  
 Người dùng và nhà phát triển có mục tiêu rõ ràng trong khi thực hiện thử nghiệm.**

* **Kiểm tra hộp xám chủ yếu được thực hiện bởi quan điểm của người dùng.**
* **Người kiểm thử không bắt buộc phải có kỹ năng lập trình cao cho thử nghiệm này.**
* **Bằng cách thực hiện kiểm tra hộp xám, lợi ích của cả thử nghiệm hộp đen và hộp trắng đều thu được**
* **Thử nghiệm hộp màu xám là không thiên vị. Nó tránh xung đột giữa người thử nghiệm và nhà phát triển  
    
  Nhược điểm của kiểm tra hộp xám**
* **Quyền truy cập hạn chế vào cấu trúc nội bộ dẫn đến quyền truy cập hạn chế cho việc truyền đường dẫn mã.**
* **Bởi vì mã nguồn không thể được truy cập, làm kiểm tra hộp trắng hoàn chỉnh là không thể.**
* **Kiểm thử hộp xám không phù hợp để kiểm tra thuật toán.**
* **Hầu hết các trường hợp kiểm thử đều khó thiết kế.**

### **Vòng đời lỗi**

****Vậy thì không muốn đối mặt với thử thách này.

#### Trước tiên chúng ta phải hiểu lỗi / khiếm khuyết là gì?

#### Một lập trình viên có thể mắc lỗi hoặc lỗi khi thiết kế và xây dựng phần mềm.

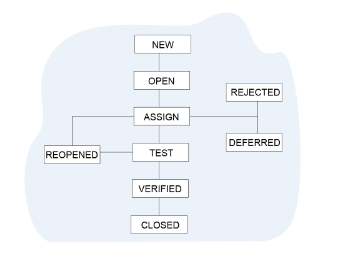
#### Những lỗi hoặc sai lầm như vậy có nghĩa là chương trình có sai sót.

#### Vòng đời khiếm khuyết hoặc Vòng đời lỗi là hành trình của một chu kỳ khiếm khuyết, mà một khiếm khuyết trải qua trong suốt vòng đời của nó.

#### Nó thay đổi từ tổ chức này sang tổ chức khác và cũng từ dự án này sang dự án khác vì nó được điều chỉnh bởi quá trình kiểm thử phần mềm.

#### Mục đích của vòng đời lỗi là tổ chức thay đổi trạng thái lỗi cho những người được gán khác nhau một cách dễ dàng và làm cho quá trình sửa lỗi có hệ thống.

### **Vòng đời khiếm khuyết / lỗi**

****Các trạng thái vòng đời khuyết tật là gì:

#### **Mới** - Khiếm khuyết tiềm ẩn được nêu ra và chưa được xác nhận.

#### Được giao - **Được giao** cho một nhóm phát triển để giải quyết nó nhưng chưa được giải quyết.

#### **Hoạt động** - Khiếm khuyết đang được nhà phát triển giải quyết và cuộc điều tra đang được tiến hành viz - Hoãn lại hoặc Từ chối.

### **Tiếp tục vòng đời lỗi**

#### **Kiểm tra** - Khiếm khuyết đã được sửa chữa và sẵn sàng để thử nghiệm.

#### Đã xác minh - Khiếm khuyết được kiểm tra lại và thử nghiệm đã được QA **xác minh**.

#### **Đóng -** Trạng thái cuối cùng của lỗi có thể được đóng lại sau khi kiểm tra lại QA hoặc có thể được đóng lại nếu lỗi trùng lặp hoặc được coi là KHÔNG phải là lỗi.

#### **Mở** lại - Khi lỗi KHÔNG được sửa, QA sẽ mở lại / kích hoạt lại lỗi.

#### **Trì** hoãn - Khi một khiếm khuyết không thể được giải quyết trong chu kỳ cụ thể đó, nó sẽ được hoãn lại để phát hành trong tương lai.

#### **Bị từ chối** - Một khiếm khuyết có thể bị từ chối vì bất kỳ lý do nào trong 3 lý do; viz - lỗi trùng lặp, KHÔNG phải là khiếm khuyết, không thể tái tạo.

### **Quản lý cấu hình**

****SCM là một quá trình để quản lý, tổ chức và kiểm soát một cách có hệ thống các thay đổi trong tài liệu, mã và các thực thể khác trong Vòng đời phát triển phần mềm.

#### Mục tiêu chính là tăng năng suất với những sai lầm tối thiểu.

#### Một chương trình CM hoàn chỉnh bao gồm các điều khoản cho việc lưu trữ, theo dõi và cập nhật tất cả thông tin hệ thống trên cơ sở thành phần và hệ thống.

### **Ví dụ**

#### Một ví dụ sẽ là tuyệt vời để hoàn thành chủ đề này:

#### Giả sử công ty của bạn phát triển một hệ thống máy bay của bên thứ ba bao gồm màn hình LCD.

#### Màn hình LCD cụ thể trở nên dư thừa theo thời gian và bạn cần tìm một sự thay thế phù hợp để tiếp tục phát triển.

#### Chính sách CM sẽ xác định ai được phép phê duyệt trong tổ chức nào sử dụng phiên bản khác, vẫn tương thích, của màn hình LCD gốc.

### **Quy trình CM**

****5 quy trình kỷ luật này của CM:

1. **Lập kế hoạch và quản lý: Hướng dẫn thiết kế sản phẩm, ai có trách nhiệm gì và bao gồm những công cụ và quy trình sử dụng. Nó xác định các yêu cầu của bên thứ ba và quá trình kiểm toán và xem xét.**
2. **Nhận dạng: Cho phép thiết lập đường cơ sở và xác định cấu hình của tài sản và / hoặc phần mềm.**
3. **Kiểm soát: Duy trì kiểm soát cấu hình, yêu cầu thay đổi và thực hiện các thay đổi đối với hệ thống và tài liệu của nó.**
4. **Kế toán trạng thái: Quá trình ghi lại và báo cáo mô tả mục cấu hình và thay đổi từ đường cơ sở trong vòng đời của mục.**
5. **Xác minh / Kiểm toán: Có thể kiểm tra những gì bạn đã giới thiệu và kiểm soát tốt tất cả các sản phẩm được quản lý**

### **Lợi ích của CM**

* **Tăng hiệu quả, ổn định và kiểm soát bằng cách cải thiện khả năng hiển thị và theo dõi.**
* **Nó rất hữu ích để tạo ra phần mềm mới với các thành phần cũ.**
* **Nó rất hữu ích để tạo ra phần mềm mới với các thành phần cũ.**
* **Xác định và áp dụng các chính sách và thủ tục có cấu trúc quy định về ghi nhận tài sản, kiểm soát trạng thái và kiểm toán.**