**Software Process**

*🟑 4 Hoạt động cơ bản cho Kỹ thuật phần mềm*

**1. Đặc điểm kỹ thuật phần mềm**

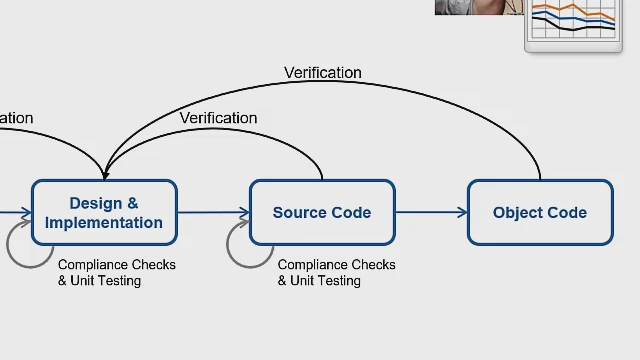
Chức năng của phần mềm và các ràng buộc trong hoạt động của nó phải được xác định.

**2. Thiết kế & triển khai phần mềm**

Phần mềm để đáp ứng các đặc điểm kỹ thuật phải được sản xuất

**3. Xác nhận phần mềm**

Phần mềm phải được xác thực để đảm bảo rằng phần mềm đó thực hiện những gì khách hàng muốn.



**4. Tiến hóa phần mềm**

Phần mềm phải phát triển để đáp ứng nhu cầu thay đổi của khách hàng.

Trong một số hình thức, các hoạt động này là một phần của tất cả các quy trình phần mềm.

Trong thực tế, tất nhiên, chúng là những hoạt động phức tạp trong bản thân và bao gồm các hoạt động phụ như xác nhận yêu cầu, thiết kế kiến trúc, thử nghiệm đơn vị, v.v.

Ngoài ra còn có các hoạt động hỗ trợ quy trình như quản lý cấu hình tài liệu và phần mềm

Khi chúng tôi mô tả và thảo luận các quy trình, chúng tôi thường nói về các hoạt động trong các quy trình này như chỉ định mô hình dữ liệu, thiết kế giao diện người dùng, v.v. và thứ tự của các hoạt động này.

Tuy nhiên, cũng như các hoạt động, mô tả quy trình cũng có thể bao gồm:

Các sản phẩm

Đó là kết quả của một hoạt động quy trình.

Ví dụ, kết quả của hoạt động thiết kế kiến trúc có thể là một mô hình của kiến trúc phần mềm.

Vai trò

Phản ánh trách nhiệm của những người liên quan đến quy trình.

Ví dụ về vai trò là người quản lý dự án, người quản lý cấu hình, lập trình viên, v.v.

Trước và sau điều kiện

Đó là những tuyên bố đúng trước và sau khi một hoạt động quy trình đã được ban hành hoặc một sản phẩm được sản xuất.

Ví dụ, trước khi thiết kế kiến trúc bắt đầu, một điều kiện trước có thể là tất cả các yêu cầu đã được khách hàng chấp thuận; sau khi hoạt động này kết thúc, một điều kiện sau có thể là các mô hình UML mô tả kiến trúc đã được xem xét.

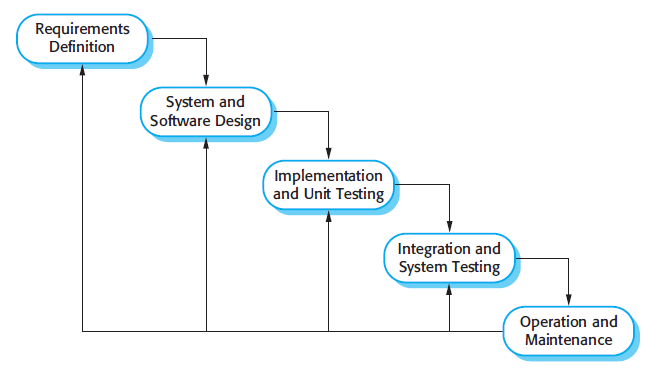
**Software Process Model**

**Mô hình tuyến tính**

Các mô hình tuyến tính phù hợp cho các dự án mà tất cả các yêu cầu được xác định và hiểu rõ trước khi thiết kế phần mềm bắt đầu.

Hai loại mô hình tuyến tính là:

**Mô hình thác nước(Waterfall model)**



Mô hình xuất bản đầu tiên của quá trình phát triển phần mềm được bắt nguồn từ các quy trình kỹ thuật hệ thống tổng quát hơn (Royce, 1970).

Do thác từ giai đoạn này sang pha khác, mô hình này được gọi là "mô hình thác nước" hoặc vòng đời phần mềm.

Mô hình thác nước là một ví dụ về quy trình định hướng kế hoạch — về nguyên tắc, bạn phải lập kế hoạch và lên lịch cho tất cả các hoạt động của quá trình trước khi bắt đầu công việc trên chúng.

*Các giai đoạn chính của mô hình thác nước*

**Yêu cầu Phân tích & Định nghĩa**

Các dịch vụ, ràng buộc và mục tiêu của hệ thống được thiết lập bằng cách tham vấn với người dùng hệ thống.

Sau đó, chúng được xác định chi tiết và phục vụ như một đặc tả hệ thống.

**Thiết kế hệ thống và phần mềm**

Quy trình thiết kế hệ thống phân bổ các yêu cầu cho các hệ thống phần cứng hoặc phần mềm bằng cách thiết lập một kiến ​​trúc hệ thống tổng thể.

Thiết kế phần mềm liên quan đến việc xác định và mô tả trừu tượng hệ thống phần mềm cơ bản và các mối quan hệ của chúng

**Thực hiện và kiểm tra đơn vị**

Trong giai đoạn này, thiết kế phần mềm được thực hiện như một tập hợp các chương trình hoặc các đơn vị chương trình. Kiểm tra đơn vị liên quan đến việc xác minh rằng mỗi đơn vị đáp ứng đặc điểm kỹ thuật của nó.

**Tích hợp và kiểm tra hệ thống**

Các chương trình hoặc chương trình riêng lẻ được tích hợp và thử nghiệm như một hệ thống hoàn chỉnh để đảm bảo rằng các yêu cầu phần mềm đã được đáp ứng.

Sau khi thử nghiệm, hệ thống phần mềm được phân phối cho khách hàng.

**Vận hành và bảo trì**

Đây là giai đoạn vòng đời dài nhất.

Hệ thống được lắp đặt và đưa vào sử dụng thực tế.

Việc bảo trì liên quan đến sửa lỗi không được phát hiện trong các giai đoạn trước của vòng đời, cải thiện việc triển khai các đơn vị hệ thống và tăng cường các dịch vụ của hệ thống khi các yêu cầu mới được phát hiện.

Mô hình thác được sử dụng trong các dự án nhỏ, nơi các yêu cầu được xác định ngay từ đầu.

Mô hình này giả định rằng các yêu cầu được ổn định và đóng băng trong kế hoạch dự án.

Tuy nhiên, điều này thường không đúng trong trường hợp các dự án lớn, nơi mà các yêu cầu có thể phát triển trong suốt quá trình phát triển.

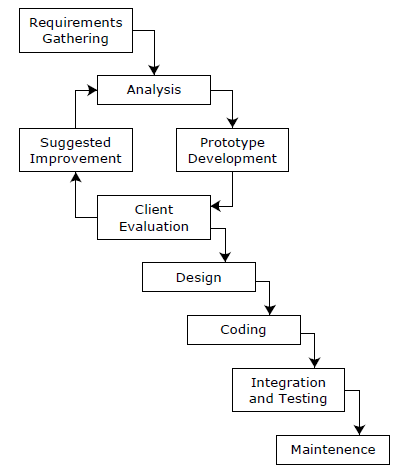
**Tạo mẫu(Prototyping Model)**

Mô hình Prototyping, còn được gọi là mô hình tiến hóa, đã đi vào hình ảnh vì những thất bại xảy ra trong phiên bản cuối cùng của ứng dụng phần mềm được phát triển bằng cách sử dụng mô hình thác nước.

Những thất bại thường xảy ra do sự thay đổi trong các yêu cầu của hệ thống được đề xuất hoặc do khoảng cách trong việc hiểu các yêu cầu của khách hàng bởi nhóm phát triển.

Một khoảng trống trong phiên bản đầu tiên của phần mềm ứng dụng được phát triển, chắc chắn, dẫn đến sự cần thiết phải làm lại ứng dụng.

Để khắc phục những hạn chế này, khái niệm tạo mẫu đã được giới thiệu.



Một hạn chế của mô hình Prototyping là nhóm phát triển đòi hỏi một chuyên gia cao cấp để phát triển một nguyên mẫu.

Nếu nhóm phát triển không đủ kinh nghiệm, nó có thể thỏa hiệp về chất lượng của hệ thống để có được nguyên mẫu hoạt động trong thời gian ngắn.

**Mô hình lặp đi lặp lại(Iterative Models)**

Các mô hình lặp đi lặp lại, một cải tiến so với mô hình Prototyping

Được sử dụng khi các yêu cầu cho phần mềm có khả năng phát triển trong suốt quá trình phát triển.

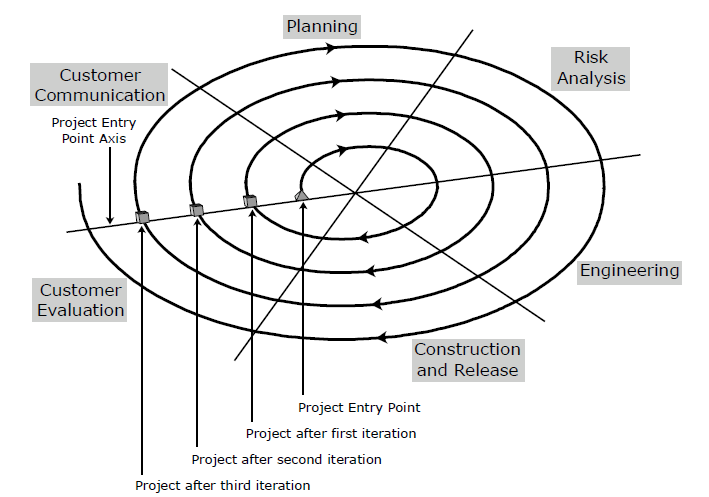
Các loại mô hình lặp lại là:

* Mô hình xoắn ốc(The Spiral Model)
* Mô hình xoắn ốc Win-win
* Mô hình phát triển dựa trên thành phần

**The Spiral Model**

Mô hình Xoắn ốc bao gồm bản chất lặp của mô hình Prototyping và bản chất tuyến tính của mô hình Thác.

Mô hình này lý tưởng cho việc phát triển phần mềm được phát hành trong các phiên bản khác nhau.



Trong mỗi lần lặp của mô hình Xoắn ốc, quy trình phát triển phần mềm tuân theo phương pháp tuyến tính phasewise.

Vào cuối lần lặp đầu tiên, khách hàng đánh giá phần mềm và cung cấp phản hồi.

Dựa trên đó, quy trình phát triển phần mềm đi vào bước lặp tiếp theo và sau đó theo cách tiếp cận tuyến tính để thực hiện phản hồi.

Quá trình lặp lại tiếp tục trong suốt vòng đời của phần mềm.

Một ví dụ về mô hình Xoắn ốc là sự phát triển của hệ điều hành Windows từ Window 3.1 đến Windows 10.

Microsoft đã phát triển hệ điều hành Windows 3.1 làm phiên bản đầu tiên của hệ điều hành.

Bạn có thể gọi đây là lần lặp đầu tiên trong mô hình Xoắn ốc. Sản phẩm được phát hành và đánh giá bởi khách hàng, trong đó bao gồm thị trường lớn.

Sau khi nhận được phản hồi từ khách hàng về Windows 3.1, Microsoft đã lên kế hoạch phát triển một phiên bản mới của hệ điều hành Windows.

Windows 95 được phát hành với những cải tiến và tính linh hoạt đồ họa. Tương tự như vậy, các phiên bản khác của hệ điều hành Windows đã được phát hành.

*6 giai đoạn của mô hình xoắn ốc*

***1.Giao tiếp khách hàng***

Bao gồm việc hiểu các yêu cầu hệ thống bằng cách liên lạc liên tục giữa khách hàng và nhà phân tích hệ thống.

***2.Lập kế hoạch***

Bao gồm ước tính lịch biểu, chi phí và tài nguyên cho dự án

***3.Phân tích rủi ro***

Bao gồm việc xác định, ước tính và theo dõi các rủi ro kỹ thuật và quản lý, chẳng hạn như trượt lịch và chi phí vượt quá.

***4.Kỹ thuật***

Bao gồm thu thập yêu cầu và thiết kế của hệ thống phần mềm

***5.Xây dựng và phát hành***

Bao gồm mã hóa, kiểm tra và triển khai phần mềm tại trang web của khách hàng và cung cấp tài liệu hỗ trợ người dùng.

***6.Đánh giá khách hàng***

Bao gồm đánh giá phần mềm của khách hàng và thực hiện phản hồi trong lần lặp tiếp theo về phát triển phần mềm

**Component-Based Development Model**

Mô hình Phát triển dựa trên thành phần nhấn mạnh rộng rãi vào khả năng sử dụng lại của các thành phần.

Một thành phần là một phần có thể nhận dạng của một chương trình lớn hơn. Nó có thể là một hàm cụ thể hoặc một nhóm các hàm liên quan.

Trong mô hình Phát triển dựa trên thành phần, một hệ thống được chia thành các thành phần, lần lượt, được tạo thành từ các mô-đun.

Bạn có thể phát triển từng thành phần của một hệ thống phần mềm bằng cách sử dụng phương pháp tuyến tính hoặc lặp lại.

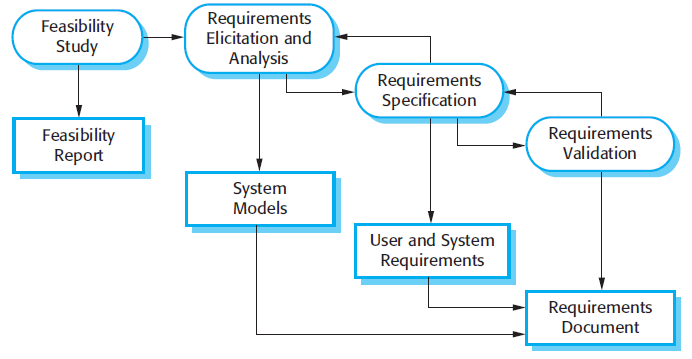
Tất cả các mô-đun liên quan tạo thành một thành phần được kiểm tra để đảm bảo rằng chúng hoạt động cùng nhau.

Bạn có thể tái sử dụng một thành phần và kết hợp nó với các thành phần khác trong cùng một hoặc các máy tính khác nhau trong một mạng phân phối để tạo thành một ứng dụng.

Ví dụ về các thành phần có thể bao gồm một máy tính lãi suất hiện có và thành phần tạo phần trả góp có thể được cắm vào phần mềm ứng dụng ngân hàng.

Tùy thuộc vào loại và số tiền cho vay và thời hạn mà khoản vay được yêu cầu, thành phần tính lãi suất tính lãi suất và thành phần tạo phần trả góp tạo ra gói trả góp theo chính sách ngân hàng.

**Process Activities**



**Đặc điểm phần mềm**

Đặc điểm kỹ thuật phần mềm hoặc kỹ thuật yêu cầu là quá trình hiểu và xác định các dịch vụ nào được yêu cầu từ hệ thống và xác định các ràng buộc về hoạt động và phát triển của hệ thống.

Yêu cầu kỹ thuật là một giai đoạn đặc biệt quan trọng của quá trình phần mềm như các lỗi ở giai đoạn này chắc chắn dẫn đến các vấn đề sau này trong việc thiết kế và thực hiện hệ thống

**Có 4 hoạt động chính trong quy trình kỹ thuật yêu cầu**

*Nghiên cứu khả thi*

Một ước tính được thực hiện cho dù nhu cầu người dùng được xác định có thể được thỏa mãn bằng cách sử dụng các công nghệ phần mềm và phần cứng hiện tại hay không.

Nghiên cứu xem xét liệu hệ thống được đề xuất sẽ có hiệu quả về mặt chi phí từ quan điểm kinh doanh hay không và liệu nó có thể được phát triển trong các ràng buộc ngân sách hiện có hay không.

Một nghiên cứu khả thi nên tương đối rẻ và nhanh chóng. Kết quả sẽ thông báo quyết định có nên tiến hành phân tích chi tiết hơn hay không

*Yêu cầu Phân tích & Phân tích*

Đây là quá trình tạo ra các yêu cầu hệ thống thông qua việc quan sát các hệ thống hiện có, các cuộc thảo luận với người dùng tiềm năng và người quản lý, phân tích nhiệm vụ, v.v.

Điều này có thể liên quan đến sự phát triển của một hoặc nhiều mô hình hệ thống và nguyên mẫu. Những điều này giúp bạn hiểu được hệ thống được chỉ định.

*Yêu cầu kỹ thuật*

Yêu cầu đặc điểm kỹ thuật là hoạt động dịch các thông tin được thu thập trong hoạt động phân tích thành một tài liệu xác định một tập hợp các yêu cầu

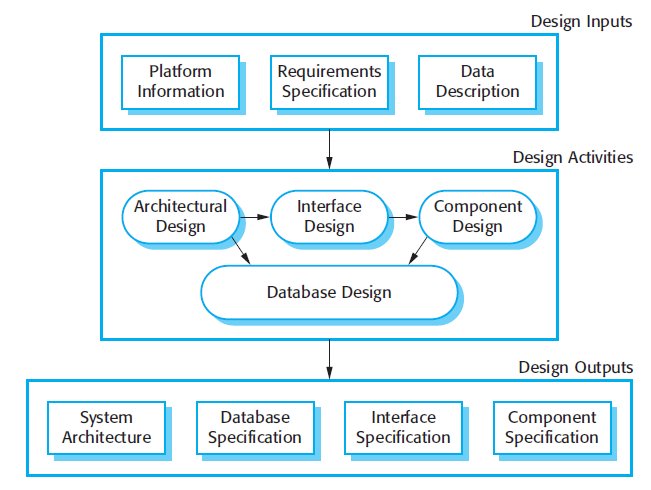
Hai loại yêu cầu có thể được bao gồm trong tài liệu này. Các yêu cầu của người dùng là các câu lệnh trừu tượng về các yêu cầu hệ thống cho khách hàng và người dùng cuối của hệ thống; yêu cầu hệ thống là mô tả chi tiết hơn về chức năng được cung cấp.

*Xác nhận yêu cầu*

Hoạt động này kiểm tra các yêu cầu về chủ nghĩa hiện thực, nhất quán và đầy đủ.

Trong quá trình này, các lỗi trong tài liệu yêu cầu chắc chắn được phát hiện. Sau đó nó phải được sửa đổi để sửa những vấn đề này.

**Software Design & Implementation**



Giai đoạn thực hiện phát triển phần mềm là quá trình chuyển đổi một đặc tả hệ thống thành một hệ thống thực thi.

Nó luôn luôn liên quan đến các quá trình thiết kế và lập trình phần mềm nhưng, nếu một cách tiếp cận gia tăng để phát triển được sử dụng, cũng có thể liên quan đến việc tinh chỉnh đặc tả phần mềm

Thiết kế phần mềm là mô tả cấu trúc của phần mềm được triển khai, các mô hình dữ liệu và cấu trúc được hệ thống sử dụng, các giao diện giữa các thành phần hệ thống và đôi khi, các thuật toán được sử dụng.

Các nhà thiết kế không đạt được thiết kế đã hoàn thành ngay lập tức mà phát triển thiết kế lặp đi lặp lại.

Họ thêm hình thức và chi tiết khi họ phát triển thiết kế của họ với backtracking liên tục để sửa thiết kế trước đó.

Hầu hết các giao diện phần mềm với các hệ thống phần mềm khác. Chúng bao gồm hệ điều hành, cơ sở dữ liệu, phần mềm trung gian và các hệ thống ứng dụng khác. Chúng tạo nên 'nền tảng phần mềm', môi trường mà phần mềm sẽ thực hiện.

Thông tin về nền tảng này là một đầu vào thiết yếu cho quá trình thiết kế, vì các nhà thiết kế phải quyết định cách tích hợp nó tốt nhất với môi trường của phần mềm

Các đặc tả yêu cầu là một mô tả các chức năng phần mềm phải cung cấp và hiệu suất và độ tin cậy yêu cầu của nó.

Nếu hệ thống xử lý dữ liệu hiện có, thì mô tả dữ liệu đó có thể được bao gồm trong đặc tả nền tảng; nếu không, mô tả dữ liệu phải là một đầu vào cho quá trình thiết kế để tổ chức hệ thống dữ liệu được xác định.

**4 Activities may be part of the design process**

*Thiết kế kiến ​​trúc*

Nơi bạn xác định cấu trúc tổng thể của hệ thống, các thành phần chính (đôi khi được gọi là các hệ thống con hoặc mô-đun), mối quan hệ của chúng và cách chúng được phân phối.

*Thiết kế giao diện*

Nơi bạn xác định giao diện giữa các thành phần hệ thống.

*Đặc tả giao diện này phải rõ ràng.*

Với giao diện chính xác, một thành phần có thể được sử dụng mà không cần các thành phần khác phải biết cách triển khai nó.

Khi các đặc tả giao diện được thỏa thuận, các thành phần có thể được thiết kế và phát triển đồng thời.

*Thiết kế thành phần*

Nơi bạn lấy từng thành phần hệ thống và thiết kế nó sẽ hoạt động như thế nào.

Đây có thể là một tuyên bố đơn giản về chức năng dự kiến ​​sẽ được thực hiện, với thiết kế cụ thể còn lại cho lập trình viên.

Ngoài ra, nó có thể là một danh sách các thay đổi được thực hiện cho một thành phần tái sử dụng hoặc một mô hình thiết kế chi tiết.

Mô hình thiết kế có thể được sử dụng để tự động tạo ra một triển khai thực hiện.

*Thiết kế cơ sở dữ liệu*

Nơi bạn thiết kế cấu trúc dữ liệu hệ thống và cách chúng được biểu diễn trong cơ sở dữ liệu.

Một lần nữa, công việc ở đây phụ thuộc vào việc một cơ sở dữ liệu hiện có có được tái sử dụng hay một cơ sở dữ liệu mới sẽ được tạo ra hay không.

Các hoạt động này dẫn đến một tập hợp các đầu ra thiết kế, cũng được hiển thị trong hình trước

Các chi tiết và đại diện của những thay đổi đáng kể. Đối với các hệ thống quan trọng, các tài liệu thiết kế chi tiết trình bày các mô tả chính xác và chính xác của hệ thống phải được tạo ra

Sự phát triển của một chương trình để thực hiện hệ thống sau một cách tự nhiên từ các quy trình thiết kế hệ thống.

Mặc dù một số lớp của chương trình, chẳng hạn như hệ thống an toàn quan trọng, thường được thiết kế chi tiết trước khi thực hiện bất kỳ bắt đầu, nó là phổ biến hơn cho các giai đoạn sau của thiết kế và phát triển chương trình được xen kẽ.

Các công cụ phát triển phần mềm có thể được sử dụng để tạo ra một chương trình khung sườn từ một thiết kế. Điều này bao gồm mã để xác định và thực hiện các giao diện, và trong nhiều trường hợp, nhà phát triển chỉ cần thêm chi tiết về hoạt động của từng thành phần chương trình

Lập trình là một hoạt động cá nhân và không có quy trình chung thường được tuân theo.

Một số lập trình viên bắt đầu với các thành phần mà họ hiểu, phát triển chúng, và sau đó chuyển sang các thành phần ít được hiểu hơn.

Những người khác có cách tiếp cận ngược lại, để lại các thành phần quen thuộc cho đến cuối cùng bởi vì họ biết làm thế nào để phát triển chúng.

Một số nhà phát triển muốn xác định dữ liệu sớm trong quá trình, sau đó sử dụng dữ liệu này để thúc đẩy phát triển chương trình; những người khác để dữ liệu không được chỉ định càng lâu càng tốt.

Thông thường, các lập trình viên thực hiện một số kiểm tra mã họ đã phát triển.

Điều này thường cho thấy các lỗi chương trình phải được loại bỏ khỏi chương trình.

Điều này được gọi là gỡ lỗi. Kiểm tra lỗi và gỡ lỗi là các quy trình khác nhau.

Kiểm tra thiết lập sự tồn tại của các khuyết tật. Gỡ lỗi có liên quan với việc định vị và sửa chữa các lỗi này.

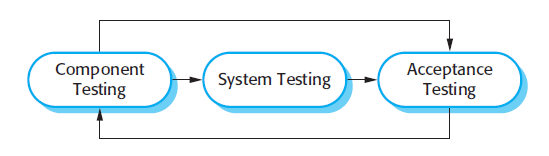
**Software Validation**

Xác nhận phần mềm hoặc, nói chung hơn, xác minh và xác thực (V & V) được dự định để cho thấy rằng một hệ thống vừa tuân theo đặc điểm kỹ thuật của nó và đáp ứng được sự mong đợi của khách hàng hệ thống

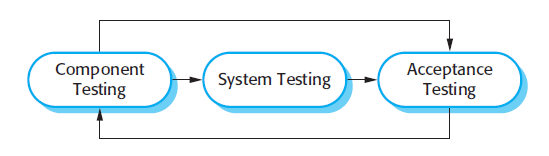
Kiểm thử chương trình, nơi hệ thống được thực hiện bằng cách sử dụng dữ liệu thử nghiệm mô phỏng, là kỹ thuật xác thực chính

Xác nhận cũng có thể bao gồm:

* Kiểm tra quá trình (như kiểm tra)
* Kiểm tra yêu cầu người dùng



**Quy trình thử nghiệm ba giai đoạn**



*Lý tưởng nhất là….*

Lỗi thành phần được phát hiện sớm trong quá trình và

Các vấn đề về giao diện được tìm thấy khi hệ thống được tích hợp

*Tuy nhiên….*

Khi các khiếm khuyết được phát hiện, chương trình phải được gỡ lỗi và điều này có thể yêu cầu các giai đoạn khác trong quá trình thử nghiệm được lặp lại.

Lỗi trong các thành phần chương trình có thể xuất hiện trong quá trình kiểm tra hệ thống. Do đó, quá trình này là một quá trình lặp đi lặp lại với thông tin được cung cấp từ các giai đoạn sau đến các phần trước của quy trình.

**Các giai đoạn trong quá trình thử nghiệm**

***Kiểm tra phát triển***

Các thành phần được kiểm tra bởi những người phát triển hệ thống

Mỗi thành phần được kiểm tra độc lập

Các thành phần có thể là các thực thể đơn giản như các hàm hoặc các lớp đối tượng

Các công cụ tự động hóa thử nghiệm như JUnit (Massol và Husted, 2003) có thể chạy lại các thử nghiệm thành phần khi các phiên bản mới của thành phần được tạo ra, thường được sử dụng

***Thử nghiệm hệ thống***

Các thành phần hệ thống được tích hợp để tạo ra một hệ thống hoàn chỉnh

Quá trình này liên quan đến việc tìm ra các lỗi do các tương tác không lường trước được giữa các thành phần và các vấn đề về giao diện thành phần.

Nó cũng quan tâm đến việc cho thấy rằng hệ thống đáp ứng các yêu cầu chức năng và phi chức năng của nó, và kiểm tra các thuộc tính hệ thống mới nổi.

Đối với các hệ thống lớn, đây có thể là một quá trình nhiều giai đoạn mà các thành phần được tích hợp để tạo thành các hệ thống con được kiểm tra riêng trước khi các hệ thống con này được tích hợp để tạo thành hệ thống cuối cùng.

***Kiểm tra chấp nhận***

Đây là giai đoạn cuối cùng trong quá trình thử nghiệm trước khi hệ thống được chấp nhận để sử dụng.

Hệ thống được kiểm tra với dữ liệu do khách hàng hệ thống cung cấp chứ không phải với dữ liệu thử nghiệm mô phỏng.

Kiểm tra chấp nhận có thể tiết lộ lỗi và thiếu sót trong định nghĩa yêu cầu hệ thống, bởi vì dữ liệu thực sự thực hiện hệ thống theo các cách khác nhau từ dữ liệu thử nghiệm.

Thử nghiệm chấp nhận cũng có thể tiết lộ các vấn đề về yêu cầu mà các cơ sở của hệ thống không thực sự đáp ứng nhu cầu của người dùng hoặc hiệu suất của hệ thống là không thể chấp nhận.

Kiểm tra chấp nhận đôi khi được gọi là 'alpha tesing'. Các hệ thống tùy chỉnh được phát triển cho một khách hàng.

Quá trình thử nghiệm alpha tiếp tục cho đến khi nhà phát triển hệ thống và khách hàng đồng ý rằng hệ thống được phân phối là việc triển khai các yêu cầu có thể chấp nhận được.

Khi một hệ thống được đưa ra thị trường như một sản phẩm phần mềm, một quá trình thử nghiệm được gọi là "beta testing" thường được sử dụng.

Thử nghiệm beta liên quan đến việc cung cấp hệ thống cho một số khách hàng tiềm năng đồng ý sử dụng hệ thống đó. Họ báo cáo vấn đề cho các nhà phát triển hệ thống

Điều này cho thấy sản phẩm sử dụng thực tế và phát hiện lỗi có thể không được dự đoán bởi các nhà xây dựng hệ thống.

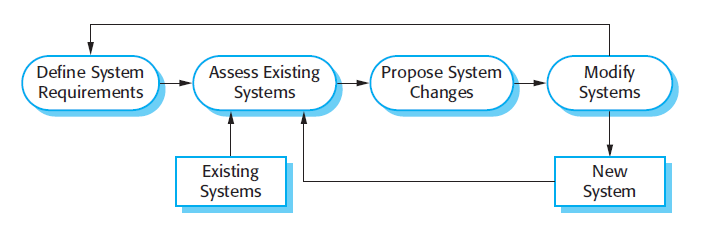
Sau phản hồi này, hệ thống được sửa đổi và phát hành để thử nghiệm beta tiếp theo hoặc để bán chung.

**Software Evolution**

Phần mềm vốn đã linh hoạt và có thể thay đổi.

Khi yêu cầu thay đổi thông qua việc thay đổi hoàn cảnh kinh doanh, phần mềm hỗ trợ doanh nghiệp cũng phải phát triển và thay đổi.

Mặc dù đã có một ranh giới giữa phát triển và tiến hóa (bảo trì) điều này ngày càng không thích hợp vì ít và ít hệ thống hoàn toàn mới.



Kỹ thuật phần mềm như là một quá trình tiến hóa, nơi phần mềm được liên tục thay đổi trong suốt cuộc đời của nó để đáp ứng với các yêu cầu thay đổi và nhu cầu của khách hàng.

**Xác định vai trò và trách nhiệm**

Những người khác nhau tương tác với nhau trong suốt giai đoạn phân tích, phát triển và triển khai của một dự án CNTT, thực hiện nhiều vai trò khác nhau trong các tổ chức tương ứng của họ

Nói chung, những người và vai trò này có thể được phân loại thành vai trò tổ chức khách hàng và vai trò tổ chức của nhà cung cấp.



**Vai trò & trách nhiệm tại các giai đoạn SDLC**

***Giai đoạn khởi đầu***

Giai đoạn bắt đầu tập trung vào các khía cạnh "những gì và tại sao" của quá trình chuyển đổi công nghệ.

Vai trò liên quan đến giai đoạn bắt đầu từ Tổ chức khách hàng là:

* Nhóm quản lý
* Đội ngũ tư vấn

Các vai trò khác nhau liên quan đến giai đoạn bắt đầu từ tổ chức nhà cung cấp là:

* Nhân viên bán hàng và tiếp thị
* Nhà phân tích thiết kế
* Kiến trúc sư

***Giai đoạn phân tích***

Giai đoạn phân tích tập trung vào việc thu thập thông tin, xác định các yêu cầu hệ thống và xây dựng các nguyên mẫu cho việc phát hiện các yêu cầu.

Giai đoạn phân tích cũng liên quan đến việc ưu tiên các yêu cầu, tạo ra và đánh giá các lựa chọn thay thế, và xem xét các khuyến nghị với ban quản lý.

Các vai trò khác nhau liên quan đến giai đoạn phân tích là:

* Người dùng
* Người quản lý
* Quản lý kinh doanh
* Nhà phân tích thiết kế

***Giai đoạn thiết kế***

Giai đoạn thiết kế bao gồm các chi tiết của các mô-đun khác nhau và logic nội bộ của chúng.

Hoạt động thiết kế thường được chia thành hai giai đoạn riêng biệt:

* Thiết kế hệ thống
* Thiết kế chi tiết

Các vai trò khác nhau liên quan đến giai đoạn thiết kế là:

* Kiến trúc sư kiến trúc / giải pháp
* Quản lý dự án
* Người dùng
* Người quản lý

***Giai đoạn xây dựng***

Mục tiêu của giai đoạn xây dựng là dịch thiết kế của hệ thống thành một mã trong một ngôn ngữ lập trình nhất định.

Các vai trò khác nhau liên quan đến giai đoạn xây dựng là:

* Quản lý dự án
* Trưởng nhóm
* Tư vấn kỹ thuật hoặc Chuyên gia
* Lập trình viên hoặc nhà phát triển

***Giai đoạn thực hiện***

Trong giai đoạn thực hiện dự án hoàn thành và giải pháp được cài đặt tại trang web của khách hàng.

Các vai trò khác nhau liên quan đến giai đoạn thực hiện là:

* Người dùng
* Người quản lý
* Quản lý dự án
* Đội ngũ lãnh đạo, lập trình viên và chuyên gia kỹ thuật

***Vai trò và năng lực***

Vai trò của cả tổ chức khách hàng và nhà cung cấp có năng lực kỹ thuật và hành vi cụ thể là:

* Nhà phân tích thiết kế
* Kiến trúc sư
* Giám đốc phát triển kinh doanh
* Lập trình viên
* Quản lý dự án
* Trưởng nhóm
* Chuyên gia kỹ thuật

***Đảm bảo chất lượng***

Để đảm bảo chất lượng trong quá trình phát triển phần mềm, bạn cần phải thực hiện các hoạt động đảm bảo chất lượng phần mềm ở từng giai đoạn của vòng đời phát triển phần mềm.

Hệ thống chất lượng thường bao gồm một bộ các nguyên tắc mô tả cấu trúc tổ chức, trách nhiệm, thủ tục, quy trình, tiêu chuẩn và định dạng để thực hiện đảm bảo chất lượng.