reviewSE

Thursday, December 14, 2023 12:41 PM

1. Phân loại sản phẩm phần mềm

- *Sản phẩm dùng chung (Generic products)*
  - Những hệ thống độc lập được chào bán trên thị trường và bất cứ ai cũng có thể mua chúng.
  - Ví dụ: các cơ sở dữ liệu, xử lý văn bản (MicrosoftOffice), các công cụ vẽ (Corel Draw, Photoshop), nhữngcông cụ quản trị dự án…
  - Người phát triển hệ thống điều khiển đặc tả của hệ thống. Sự thay đổi phần mềm là do người phát triển quyết định và thực hiện.

- *Sản phẩm đặt hàng (Customized products)*
  - Phần mềm được phát triển cho một khách hàng cụ thể để đáp ứng nhu cầu của họ.
  - Ví dụ: hệ thống điều khiển nhúng, phần mềm điều khiển không lưu, hệ thống điều khiển giao thông.
  - Người phát triển hệ thống điều khiển đặc tả của hệ thống. Sự thay đổi phần mềm là do người phát triển quyết định và thực hiện.

2. Tiêu chí của một phần mềm tốt

- Tính bảo trì được (Maintainability) :Phần mềm phải cải tiến được để đáp ứng những thay đổi về nhu cầu của khách hàng.
- Tính tin cậy được (Dependability and security): Tính tin cậy của phần mềm gồm các đặc tính: độ tin cậy, an toàn và bảo mật.
- Tính hiệu quả (Efficiency): Phần mềm không nên sử dụng lãng phí các tài nguyên hệ thống.
- Tính chấp nhận được (Acceptability): Phần mềm phải được chấp nhận bởi người sử dụng.

3. Các hoạt động quy trình phần mềm

- *Đặc tả (Software specification):* Định nghĩa phần mềm sẽ sản xuất và các ràng buộc về mặt chức năng của phần mềm.
  - Là quy trình thiết lập danh sách các dịch vụ được yêu cầu và các ràng buộc đối với hoạt động của hệ thống và việc phát triển hệ thống.
  - Quy trình công nghệ yêu cầu (requirements engineering process)
    - Nghiên cứu khả thi (Feasibility study)
    - Thu thập và phân tích yêu cầu (Requirements elicitation and analysis)
    - Đặc tả yêu cầu (Requirements specification)
    - Thẩm định yêu cầu (Requirements validation)

- *Phát triển (Software development)*: Phần mềm được thiết kế và lập trình.
  - Là quy trình chuyển đổi các đặc tả thành hệ thống thực thi được.
  - Thiết kế phần mềm (software design): Thiết kế một cấu trúc phần mềm để hiện thực hóa đặc tả;
  - Cài đặt (implementation): Dịch cấu trúc đó thành chương trình thực thi được.
  - Các hoạt động của pha thiết kế và cài đặt thường liên quan đến nhau hoặc có thể đan xen nhau.

- *Thẩm định (Software validation)*: Phần mềm được kiểm tra để đảm bảo là nó đáp ứng được yêu cầu người dùng.
  - Kiểm định (verification) và thẩm định (validation) (V & V) nhằm mục đích chỉ ra rằng
    - Một hệ thống tuân theo đặc tả của nó;
    - Thỏa mãn yêu cầu của khách hàng hệ thống.
  - Bao gồm các quy trình kiểm tra, duyệt (review) và kiểm thử hệ thống (system testing).
  - Kiểm thử hệ thống bao gồm việc chạy hệ thống sử dụng các test case được viết ra dựa vào đặc tả.
  - Kiểm thử (Testing) là hoạt động V&V thường được sử dụng nhất.
  - Các giai đoạn kiểm thử:
    - Kiểm thử trong khi phát triển
    - Kiểm thử hệ thống
    - Kiểm thử người dùng

- *Cải tiến (Software evolution):* Phần mềm được thay đổi để đáp ứng được sự thay đổi yêu cầu của người dùng và yêu cầu của thị trường.
  - Phần mềm vốn dĩ linh hoạt và có thể thay đổi.
  - Yêu cầu thay đổi do hoạt động thương mại thay đổi è phần mềm hỗ trợ cũng phải cải tiến và thay đổi theo.
  - Ranh giới giữa phát triển phần mềm và cải tiến phần mềm ngày càng mờ nhạt đi. Ngày càng ít phần mềm được phát triển hoàn toàn mới.

4. Những nguyên tắc cơ bản

- Một số nguyên tắc cơ bản có thể áp dụng cho tất cả các loại phần mềm, không phân biệt các kỹ thuật phát triển được sử dụng:
    - Hệ thống nên được phát triển sử dụng quy trình phát triển dễ hiểu và có thể quản lý được.
    - Hiệu năng (performance) và độ tin cậy là quan trọng đối với tất cả các loại hệ thống.
    - Việc hiểu và quản lý được các yêu cầu và đặc tả phần mềm là quan trọng.
    - Nếu có thể, nên sử dụng lại phần mềm hơn là viết mới hoàn toàn.

5. Các vấn đề về đạo đức trong CNPM

- Ngoài các kỹ năng về kỹ thuật, CNPM còn đòi hỏi các kỹ sư các trách nhiệm.
- Các kỹ sư CNPM chuyên nghiệp phải trung thực và có trách nhiệm về mặt đạo đức.
- Các hành vi về mặt đạo đức không chỉ đơn thuần là tuân thủ luật pháp mà còn liên quan đến một tập các chuẩn mực đạo đức.

6. Trách nhiệm nghề nghiệp

- Bảo mật
- Năng lực
- Quyền sở hữu trí tuệ
- Lạm dụng máy tính

7. Quy trình hoạch định sẵn và quy trình linh hoạt

- *Các quy trình hoạch định sẵn (plan-driven process)* là các quy trình mà trong đó tất cả các hoạt động được lên kế hoạch trước và tiến độ thực hiện được đánh giá dựa vào kế hoạch này.
- *Trong các quy trình linh hoạt (agile process)*, kế hoạch được phát triển dần dần và dễ dàng thay đổi quy trình để đáp ứng sự thay đổi yêu cầu của khách hàng.
- Hầu hết các quy trình thực tế đều gồm những phần tử của cả hai phương pháp này.
- Không có quy trình phần mềm đúng hay sai!

8. Các mô hình quy trình phần mềm

- *Mô hình thác nước (waterfall model):*
    - Mô hình hoạch định sẵn. Các pha đặc tả và phát triển phân biệt và tách rời nhau.
    - Ưu điểm:
        - Quy trình rõ ràng -> người quản lý dễ dàng theo dõi tiến độ công việc.
        - Được sử dụng trong các hệ thống lớn trong đó hệ thống được phát triển tại nhiều địa điểm khác nhau. (Vì là quy trình hoạch định sẵn -> giúp cho việc phối hợp trong công việc dễ dàng hơn.)
    - Nhược điểm:
        - Khó khăn trong việc thích nghi với sự thay đổi khi quy trình đã vào guồng. (pha này hoàn thành rồi mới bắt đầu pha tiếp theo)
        - Không linh động trong việc chia dự án thành những giai đoạn tách biệt, khó đáp ứng sự thay đổi yêu cầu người dùng.
        - Mô hình này chỉ hợp lý khi yêu cầu được hiểu rõ và ít thay đổi trong suốt quá trình phát triển;
        - Hệ thống thương mại thường có yêu cầu không ổn định -> mô hình thác nước không phù hợp.

- *Mô hình phát triển dần dần (incremental development)*: Các pha đặc tả, phát triển và thẩm định đan xen nhau. Có thể là mô hình hoạch định sẵn, có thể là mô hình linh hoạt.
    - Ưu điểm
        - Giảm được chi phí khi đáp ứng sự thay đổi yêu cầu của khách hàng
        - Dễ dàng trong việc lấy phản hồi từ khách hàng.
        - Phân phối và triển khai phần mềm đến khách hàng nhanh hơn.
    - Nhược điểm
        - Quy trình không rõ ràng.
        - Cấu trúc hệ thống có xu hướng bị giảm đi vì

      những phần mới của hệ thống được thêm vào.

- *CNPM theo hướng tái sử dụng (reuse-oriented software engineering)*: Hệ thống được xây dựng từ những component có sẵn. Có thể là hoạch định sẵn, có thể là linh hoạt.
- Các giai đoạn của quy trình
    - Phân tích component;
    - Bổ sung yêu cầu;
    - Thiết kế hệ thống với việc tái sử dụng;
    - Phát triển và tích hợp.
- Hiện nay, việc tái sử dụng là phương pháp chuẩn cho việc xây dựng nhiều hệ thống thương mại.
- Thực tế, những hệ thống lớn được phát triển bằng cách sử dụng quy trình tạo ra bằng cách kết hợp các phần tử từ các mô hình này.

9. Các hoạt động quản trị

- Lên kế hoạch dự án

- Người quản trị dự án chịu trách nhiệm lên kế hoạch, ước lượng và lên lịch trình phát triển dự án và gán công việc cho thành viên của dự án.
- Viết báo cáo
  - Người quản trị dự án thường chịu trách nhiệm viết báo cáo về tiến độ dự án cho khách hàng và cho người quản lý của công ty phát triển phần mềm.
- Quản lý rủi ro
  - Người quản trị dự án phải đánh giá rủi ro có thể ảnh hưởng đến một dự án, điều khiển các rủi ro này và đưa ra giải pháp cụ thể khi có vấn đề phát sinh.
- Quản trị con người
  - Người quản trị dự án phải có trách nhiệm quản lý nhóm của mình chọn và thiết lập cách làm việc để đạt được hiệu quả cao cho nhóm.
- Viết đề xuất
  - Giai đoạn đầu tiên trong quản trị phần mềm có thể là viết một đề xuất để giành được hợp đồng để tiến hành một phần của công việc. Đề xuất mô tả mục tiêu của dự án và cách nó được tiến hành.

10. Các loại yêu cầu

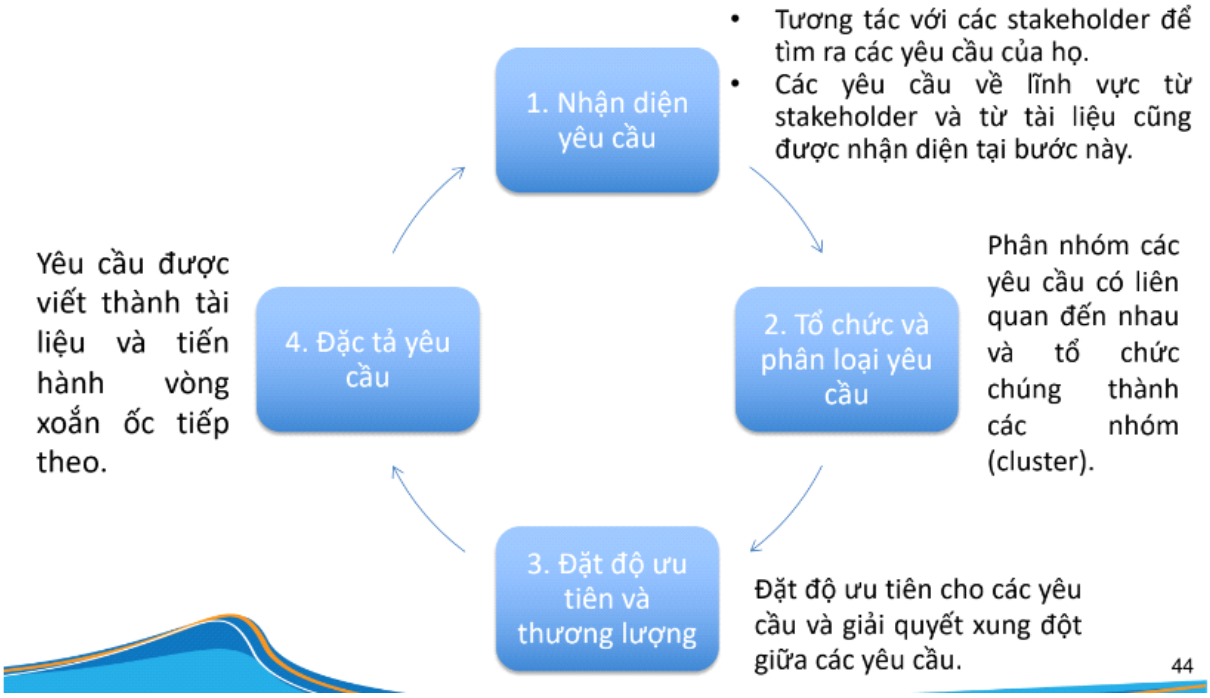
- *Yêu cầu người dùng (user requirement)*
  - Những phát biểu (bằng ngôn ngữ tự nhiên kết hợp với các biểu đồ) về các dịch vụ mà hệ thống cung cấp và những ràng buộc về hoạt động của nó.
  - Viết cho khách hàng.
- *Yêu cầu hệ thống (system requirement)*
  - Một tài liệu có cấu trúc mô tả chi tiết chức năng của hệ thống, các dịch vụ và ràng buộc về hoạt động của hệ thống.
  - Định nghĩa chính xác cái gì cần được cài đặt. Có thể là một phần của hợp đồng giữa khách hàng và người nhận thầu.

11. Yêu cầu chức năng và yêu cầu phi chức năng

- *Yêu cầu chức năng*
  - Những phát biểu về các dịch vụ mà hệ thống cung cấp, cách mà hệ thống xử lý với các đầu vào cụ thể và cách hệ thống ứng xử trong các tình huống cụ thể
  - Có thể phát biểu cả những gì mà hệ thống không làm được.
    - Mô tả chức năng và dịch vụ hệ thống cung cấp.
    - Phụ thuộc vào loại phần mềm, người sử dụng.
    - Yêu cầu chức năng người dùng là những phát biểu ở mức cao về những gì hệ thống sẽ làm.
    - Yêu cầu chức năng hệ thống mô tả các dịch vụ hệ thống ở mức chi tiết.

- *Yêu cầu phi chức năng*
  - Những ràng buộc về dịch vụ hay chức năng cung cấp bởi hệ thống như ràng buộc về thời gian, ràng buộc về quy trình phát triển, các chuẩn, ...
  - Thường áp dụng cho toàn hệ thống hơn là một chức năng hay dịch vụ đơn lẻ.
    - Xác định những thuộc tính và ràng buộc của hệ thống (độ tin cậy, thời gian trả lời và yêu cầu về mặt lưu trữ, ...)
    - Có thể quan trọng hơn yêu cầu chức năng.
      - Nếu những yêu cầu này không đạt được, hệ thống sẽ trở nên vô dụng.
  - Phân loại yêu cầu phi chức năng
    - Yêu cầu sản phẩm
      - Yêu cầu đặc tả hay ràng buộc về thuộc tính của phần mềm.
      - Ví dụ: yêu cầu về hiệu năng của phần mềm liên quan đến tốc độ thực thi, lượng bộ nhớ sử dụng, độ tin cậy, ...
    - Yêu cầu tổ chức
      - Yêu cầu xuất phát từ các chính sách và thủ tục về mặt tổ chức.
      - Ví dụ: yêu cầu về quy trình hoạt động, yêu cầu về quy trình phát triển, môi trường phát triển và chuẩn về quy trình được sử dụng...
    - Yêu cầu bên ngoài
      - Yêu cầu xuất phát từ những nhân tố bên ngoài ảnh hưởng đến hệ thống và quy trình phát triển của nó.
      - Ví dụ: yêu cầu về tương tác, yêu cầu về mặt pháp lý, …

Quy trình thu thập và phân tích yêu cầu



44

Tiêu chí kiểm tra yêu cầu

- Tính hợp lệ (Validity)
  - Hệ thống có cung cấp những chức năng đáp ứng tốt nhu cầu của người dùng ko?
- Tính nhất quán (Consistency)
  - Có các yêu cầu nào xung đột nhau hay không?
- Tính đầy đủ (Completeness)
  - Có đủ các chức năng mà khách hàng yêu cầu không?
- Tính thực tế (Realism)
  - Có thể cài đặt các yêu cầu với ngân sách và công nghệ cho trước không?
- Tính kiểm định được (Verifiability)
  - Có cách nào kiểm tra được các yêu cầu không?

Mô hình cấu trúc

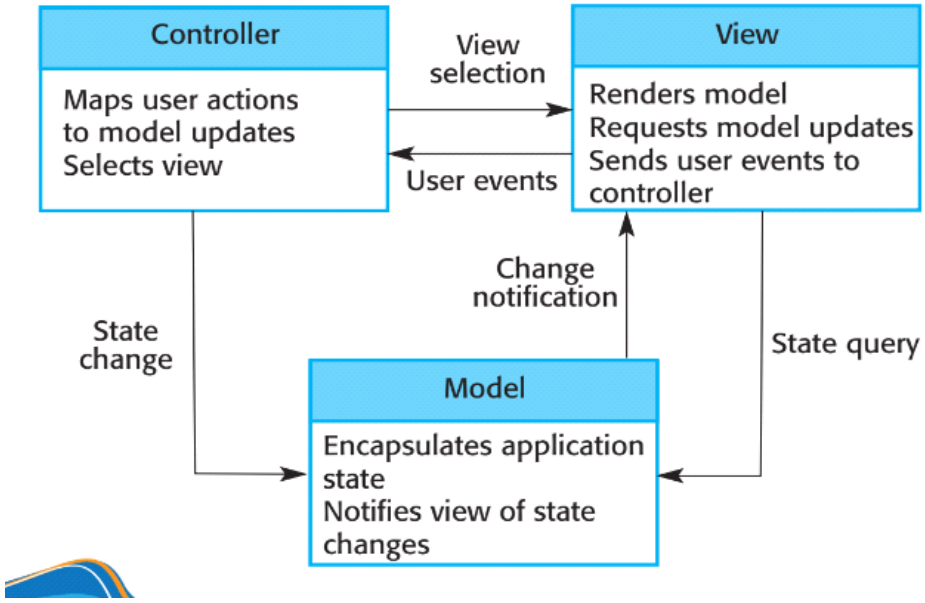
- Hiển thị cấu trúc của một hệ thống về các component tạo nên hệ thống đó và mối quan hệ của chúng.
- Các mô hình cấu trúc có thể là

Mô hình cấu trúc

- Hiển thị cấu trúc của một hệ thống về các component tạo nên hệ thống đó và mối quan hệ của chúng.
- Các mô hình cấu trúc có thể là
  - Mô hình tĩnh (static model): chỉ ra cấu trúc của thiết kế hệ thống,
  - Mô hình động (dynamic model): chỉ ra tổ chức của hệ thống khi nó được thực thi.
- Tạo ra các mô hình cấu trúc của một hệ thống khi thảo luận và thiết kế kiến trúc hệ thống.

12. Mô hình MVC

Tên	Mô hình MVC (Model-View-Controller)
Mô tả	Tách riêng phần biểu diễn và phần tương tác ra khỏi dữ liệu hệ thống. Ba component tương tác với nhau. • **Model component**: quản lý dữ liệu hệ thống và các thao tác trên dữ trên dữ liệu đó. • **View component**: định nghĩa và quản lý cách dữ liệu được biểu diễn tới người dùng như thế nào. • **Controller component**: Quản lý tương tác người dùng (ví dụ như ấn phím, nhấp chuột, ...) và chuyển các tương tác này tới View và Model.
Sử dụng khi nào	• Khi có nhiều cách biểu diễn và tương tác với dữ liệu. • Khi chưa biết được các yêu cầu tương lai cho tương tác và biểu diễn dữ liệu.
Ưu điểm	Cho phép dữ liệu thay đổi độc lập với hiển thị và ngược lại. Hỗ trợ biểu diễn theo nhiều cách khác nhau trên cùng một dữ liệu.
Nhược điểm	Có thể chứa code bổ sung và code sẽ phức tạp hơn khi mô hình dữ liệu và mô hình tương tác đơn giản.



13. Mô hình kiến trúc phân tầng

Tên	Kiến trúc phân tầng
Mô tả	Tổ chức hệ thống thành các tầng, mỗi tầng chứa các chức năng liên quan đến nhau. Một tầng cung cấp các dịch vụ cho tầng trên của nó vì vậy các tầng thấp nhất biểu diễn các dịch vụ lõi được sử dụng trong toàn bộ hệ thống.
Sử dụng khi nào	• Khi xây dựng các tính năng mới dựa trên những hệ thống có sẵn; • Khi việc phát triển được dàn trải trên nhiều nhóm khác nhau và mỗi nhóm chịu trách nhiệm về chức năng của một tầng; • Khi có yêu cầu về bảo mật ở nhiều mức độ.
Ưu điểm	• Cho phép thay thế các phần miễn là interface được duy trì. • Các chức năng dư thừa (ví dụ như phân quyền) có thể được cung cấp ở mỗi tầng để tăng độ tin cậy của hệ thống.
Nhược điểm	• Thực tế: cung cấp một sự phân chia rõ rệt giữa các tầng thường rất khó khăn và tầng cao hơn có thể tương tác trực tiếp với tầng thấp hơn hơn là thông qua một tầng bên dưới nó. • Hiệu năng cũng có thể là một vấn đề vì có nhiều mức diễn giải của một yêu cầu dịch vụ khi nó được thực hiện tại mỗi tầng.

2

14. Mô hình Repository

Tên	Mô hình Repository
Mô tả	• Tất cả các dữ liệu trong hệ thống được quản lý ở một kho trung tâm, kho này được truy cập bởi tất cả các component của hệ thống. • Các component không tương tác trực tiếp với nhau, chỉ thông qua kho chung thôi.
Sử dụng khi nào	• Khi ta có một hệ thống trong đó một lượng lớn thông tin sinh ra phải được lưu trữ trong một thời gian dài. • Sử dụng trong các hệ thống hướng dữ liệu
Ưu điểm	• Các component có thể độc lập với nhau – chúng không cần biết sự tồn tại của các component khác. Các thay đổi xảy ra ở một component không ảnh hưởng tới các component khác. • Tất cả các dữ liệu có thể được quản lý một cách nhất quán (ví dụ như backup dữ liệu được thực hiện đồng thời) vì tất cả dữ liệu được lưu trữ ở cùng một nơi.
Nhược điểm	• Các vấn đề xảy ra trên kho chung ảnh hưởng đến toàn hệ thống. • Có thể không hiệu quả trong việc tổ chức các giao tiếp thông qua kho. • Phân tán kho trên nhiều máy tính có thể khó khăn.

15. Mô hình Client - server

Tên	Mô hình client-server
Mô tả	• Chức năng của hệ thống được tổ chức thành các dịch vụ, mỗi dịch vụ được đặt trên một server riêng lẻ. • Khách hàng là người sử dụng các dịch vụ này và truy cập vào các server để sử dụng dịch vụ.
Sử dụng khi nào	• Khi dữ liệu trong một cơ sở dữ liệu chia sẻ phải truy cập từ nhiều nơi. Vì các server được truy cập từ nhiều nơi khác nhau, có thể được sử dụng khi tải trên hệ thống thay đổi.
Ưu điểm	• Server được phân tán trên mạng. • Chức năng chung (dịch vụ in ấn chẳng hạn) có thể có sẵn cho tất cả các khách hàng và không cần thiết phải cài đặt toàn bộ các dịch vụ.
Nhược điểm	• Mỗi dịch vụ là một điểm đơn gây lỗi vì vậy dễ bị tấn công từ chối dịch vụ hoặc lỗi server. • Hiệu năng có thể không dự đoán trước được do nó phụ thuộc vào mạng cũng như hệ thống. • Có thể có các vấn đề về quản lý nếu server được sở hữu bởi các tổ chức khác nhau.

16. Mô hình Pipe and filter

Tên	Mô hình pipe and filter
Mô tả	• Việc xử lý dữ liệu trong một hệ thống được tổ chức sao cho mỗi component xử lý (filter) là rời rạc và tiến hành một thao tác xử lý chuyển đổi dữ liệu. • Dòng dữ liệu (pipe) đi từ một component đến một component khác.
Sử dụng khi nào	• Trong các ứng dụng xử lý dữ liệu (cả ứng dụng xử lý khối và xử lý giao tác) trong đó các đầu vào được xử lý ở các giai đoạn rời rạc để tạo ra các đầu ra tương ứng.
Ưu điểm	• Dễ hiểu và hỗ trợ việc tái sử dụng các chuyển đổi. • Phù hợp với cấu trúc của của nhiều quy trình thương mại. • Thêm vào các chuyển đổi mới một cách dễ dàng. • Có thể cài đặt theo kiểu tuần tự hoặc song song.
Nhược điểm	• Format của dữ liệu truyền đi phải được chấp thuận trong việc giao tiếp giữa các chuyển đổi: Mỗi chuyển đổi phải phân tích cú pháp đầu vào của nó và chuyển nó thành đầu ra ở dạng được chấp nhận. → khó khăn trong việc tái sử dụng các hàm chuyển đổi khi cấu trúc dữ liệu không tương thích.

17. Các kiểu tương tác

Kiểu tương tác	Ưu điểm	Nhược điểm	Ví dụ
Thao tác trực tiếp	• Tương tác nhanh và trực quan • Dễ học	• Có thể khó cài đặt.	Video games Hệ thống CAD
Chọn menu	• Tránh lỗi người dùng • Yêu cầu gõ ký tự ít	• Thao tác chậm đối với người sử dụng có kinh nghiệm. • Có thể trở nên phức tạp nếu có nhiều lựa chọn menu.	Phần lớn các hệ thống thông dụng
Điền vào form	• Nhập dữ liệu đơn giản • Dễ học • Kiểm tra được	• Tốn nhiều không gian màn hình • Rắc rối xảy ra khi các lựa chọn của người dùng không khớp với các trường của form.	Khai thuế, xử lý nợ cá nhân
Ngôn ngữ lệnh	• Mạnh và linh động	• Khó học • Quản lý lỗi kém	Hệ điều hành, hệ thống điều khiển và lệnh
Ngôn ngữ tự nhiên	• Người sử dụng bình thường có thể dùng được. • Dễ mở rộng	• Yêu cầu gõ nhiều. • Hệ thống hiểu ngôn ngữ tự nhiên không tin cậy được	Hệ thống truy vấn thông tin

18. Các nhân tố thiết kế trong thông điệp

Nhân tố	Mô tả
Ngữ cảnh	Bất cứ khi nào có thể, hệ thống cần tạo ra các thông điệp phản ánh đúng ngữ cảnh người dùng. Hệ thống nên nhận biết được người dùng đang làm gì và nên phát sinh các thông điệp liên quan đến hoạt động hiện tại của họ.
Kinh nghiệm	Vì người dùng quen dần với hệ thống, họ sẽ trở nên khó chịu bởi các thông điệp dài dòng. Tuy nhiên, người mới dùng lại cảm thấy khó khăn để hiểu các thông báo ngắn gọn. Ta nên cung cấp cả hai loại thông điệp và cho phép người sử dụng điều khiển kiểu thông điệp họ muốn.
Kỹ năng	Thông điệp nên dựa vào kỹ năng và kinh nghiệm của người dùng. Thông điệp cho các lớp người dùng khác nhau có thể được thể hiện theo các cách khác nhau phụ thuộc vào thuật ngữ mà họ biết.
Phong cách	Thông điệp nên biểu diễn theo dạng tích cực hơn là tiêu cực, nên chủ động hơn là bị động, không bao giờ được xúc phạm hay cố gắng pha trò.
Văn hóa	Người thiết kế nên đưa ra thông điệp gần gũi với văn hóa của đất nước mà hệ thống được bán. Có nhiều sự khác biệt về văn hóa giữa châu Âu, châu Á và châu Mỹ. Một thông điệp phù hợp với văn hóa này có thể không phù hợp với văn hóa khác.

19. Kiểm định và thẩm định

- Kiểm định (verification):
  "Are we building the product right".
  - Phần mềm phải tương thích với đặc tả.
- Thẩm định(validation):
  "Are we building the right product".
  - Phần mềm phải thỏa mãn được những gì người dùng thật sự yêu cầu.

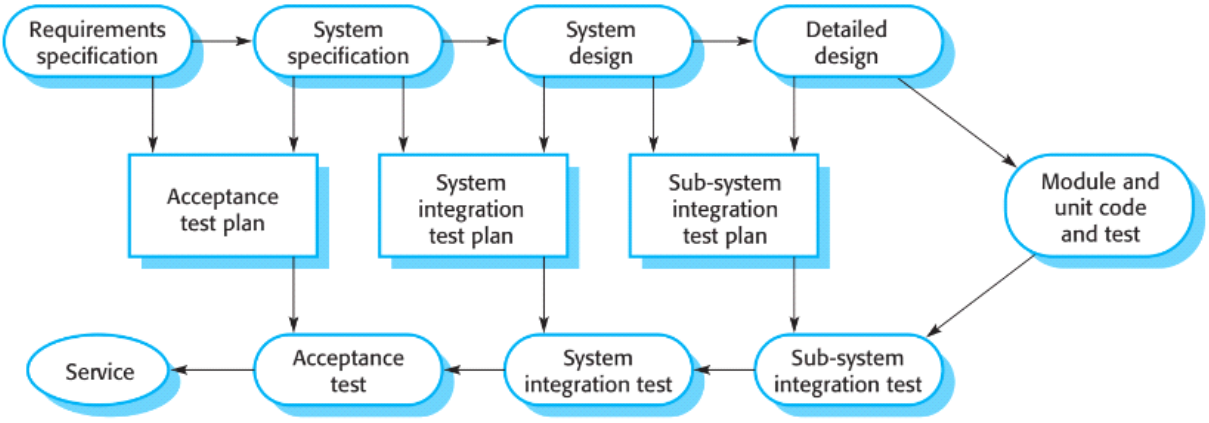
Quy trình phát triển phần mềm truyền thống:

1. Tuần tự và Tuyến tính:
   - Tuần tự: Giai đoạn phát triển được thực hiện theo trình tự tuyến tính, từ yêu cầu đến triển khai.
   - Không linh hoạt: Khó thích ứng với sự thay đổi trong yêu cầu hoặc môi trường.
2. Yêu cầu và Thiết kế Đầy Đủ Trước:
   - Yêu cầu đầy đủ: Yêu cầu phải được xác định rõ ràng từ đầu và không thay đổi nhiều.
   - Thiết kế trước: Cần thiết kế toàn bộ hệ thống1 trước khi bắt đầu việc lập trình.
3. Kiểm thử cuối cùng:
   - Cuối cùng: Kiểm thử thường được thực hiện ở giai đoạn cuối của quy trình phát triển.

Quy trình phát triển phần mềm linh hoạt:

1. Linh hoạt và Tương tác Liên Tục:
   - Phản hồi liên tục: Tương tác thường xuyên với khách hàng để hiểu rõ hơn về yêu cầu và sự thay đổi.
   - Thay đổi linh hoạt: Có khả năng thay đổi yêu cầu và thiết kế theo thời gian.
2. Scrum và Sprints:
   - Scrum: Sử dụng mô hình Scrum, chia dự án thành các đợt gọi là "Sprints" (chu kỳ phát triển ngắn).
   - Iterative và Linh hoạt: Phát triển theo phương pháp lặp và linh hoạt để có thể thích ứng với sự thay đổi.
3. Kiểm thử Liên Tục:
   - Liên tục: Kiểm thử thường xuyên được tích hợp vào quá trình phát triển hàng ngày.
   - Kiểm thử Đơn vị và Tự động: Sử dụng kiểm thử đơn vị và tự động để đảm bảo chất lượng.
4. Cộng đồng và Tự quản lý:
   - Tự quản lý: Nhóm tự quản lý và tự tổ chức để đạt được mục tiêu Sprint.
   - Cộng đồng: Tạo ra sự tương tác và hỗ trợ giữa các thành viên trong nhóm.
5. Giao Tiếp Liên Tục:
   - Giao tiếp: Giao tiếp mở cửa và liên tục giữa các thành viên của nhóm và khách hàng.
6. Phân phối Liên tục:
   - Liên tục: Phần mềm có thể được phân phối liên tục theo từng phiên bản nhỏ và có thể sử dụng sớm.
7. Phản Hồi và Điều Chỉnh Nhanh chóng:
   - Phản hồi ngay: Có khả năng phản hồi nhanh chóng từ khách hàng và thực hiện điều chỉnh ngay khi cần thiết.

V-model


Requirements specification
System specification
System design
Detailed design
Acceptance test plan
System integration test plan
Sub-system integration test plan
Module and unit code and test
Service
Acceptance test
System integration test
Sub-system integration test