CÔNG NGHỆ PHẦN MỀM

CHƯƠNG 1

Nhập môn Công nghệ phần mềm

Nghệ An, 2022

📖 Nội dung giảng dạy

1.1. Phần mềm và tầm quan trọng

1.2. Vai trò của phần mềm

1.3. Đặc trưng của phần mềm

1.4. Các tiến trình phần mềm

1.5. Một số khái niệm liên quan

📖 Mục tiêu học phần

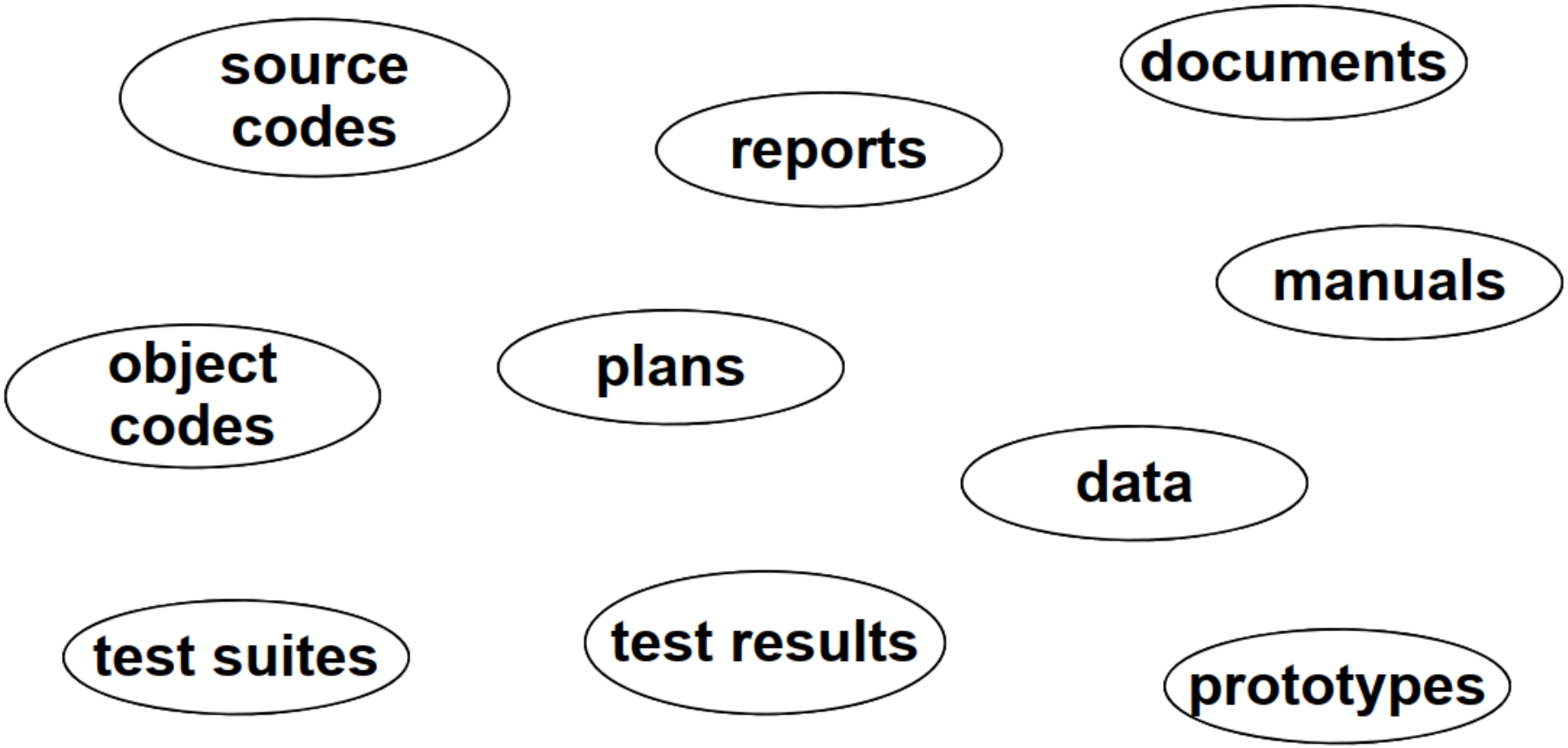
- Giúp sinh viên
  - Hiểu và giải thích được quy trình phát triển phần mềm
  - Phân tích được các yêu cầu của người sử dụng
  - Lựa chọn một mô hình quy trình phát triển phần mềm thích hợp cho một sản phẩm cụ thể.
  - Giải thích tầm quan trọng của các hoạt động đánh giá chất lượng phần mềm.
  - Biết được phải tạo ra những kết quả gì trong từng giai đoạn của quy trình phát triển phần mềm.
  - Áp dụng các mô hình thiết kế hệ thống thích hợp cho từng sản phẩm cụ thể.

1.1. Phần mềm và tầm quan trọng

- Định nghĩa phần mềm:
  - Là một tập hợp những câu lệnh được viết bằng một hoặc nhiều ngôn ngữ lập trình theo một trật tự xác định nhằm tự động thực hiện một số chức năng hoặc giải quyết một bài toán nào đó.
  - Phần mềm được thực thi trên máy, thường là máy tính.
- Phần mềm gồm chương trình máy tính, tất cả các tài liệu và dữ liệu liên quan
  - Các chương trình
  - Các tệp cấu hình
  - Các tài liệu hệ thống (các yêu cầu, mô hình thiết kế, tài liệu hướng dẫn sử dụng)
  - Websites cập nhật thông tin sản phẩm
  - …
- Phần mềm luôn gắn với một hệ thống cụ thể

1.1. Phần mềm và tầm quan trọng

- Phần mềm:



1.1. Phần mềm và tầm quan trọng

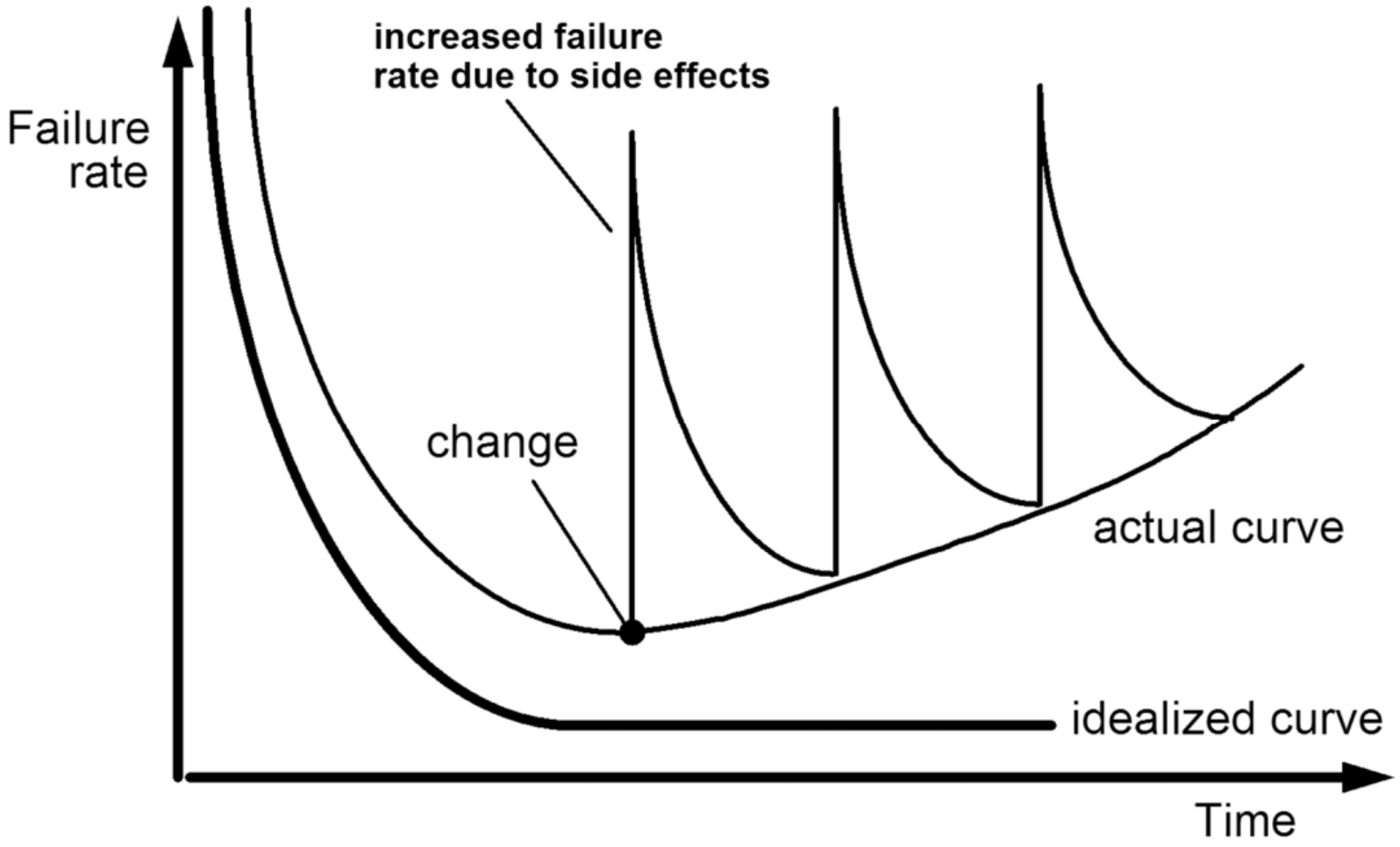
- Phần mềm:

1.1. Phần mềm và tầm quan trọng

- Đặc điểm của phầm mềm:
  - Phần mềm phải được tạo bằng cách phát triển (develop or engineer) chứ không phải đơn thuần là sản xuất (manufacture).
  - Phần mềm không hao mòn.
  - Phần mềm thì phức tạp, chi phí cho những thay đổi (change) ở những giai đoạn sau rất cao.
  - Hầu hết phần mềm vẫn phải xây dựng bằng cách tùy biến.

1.1. Phần mềm và tầm quan trọng

- Lỗi theo thời gian



1.1. Phần mềm và tầm quan trọng

- Bản chất của phần mềm:
  - Phần mềm không sờ thấy được.
  - Phần mềm dễ dàng nhân bản.
  - Phần mềm khó đánh giá về chất lượng.
  - Phần mềm dễ dàng thay đổi.

Phân loại phần mềm

- Phân loại theo phương thức hoạt động:
  - Phần mềm hệ thống: hệ điều hành, thư viện liên kết động, trình điều khiển...
  - Phần mềm ứng dụng: phần mềm văn phòng, phần mềm doanh nghiệp, phần mềm giáo dục, phần mềm giải trí...
  - Phần mềm chuyển dịch mã: trình biên dịch, trình thông dịch.

Phân loại phần mềm

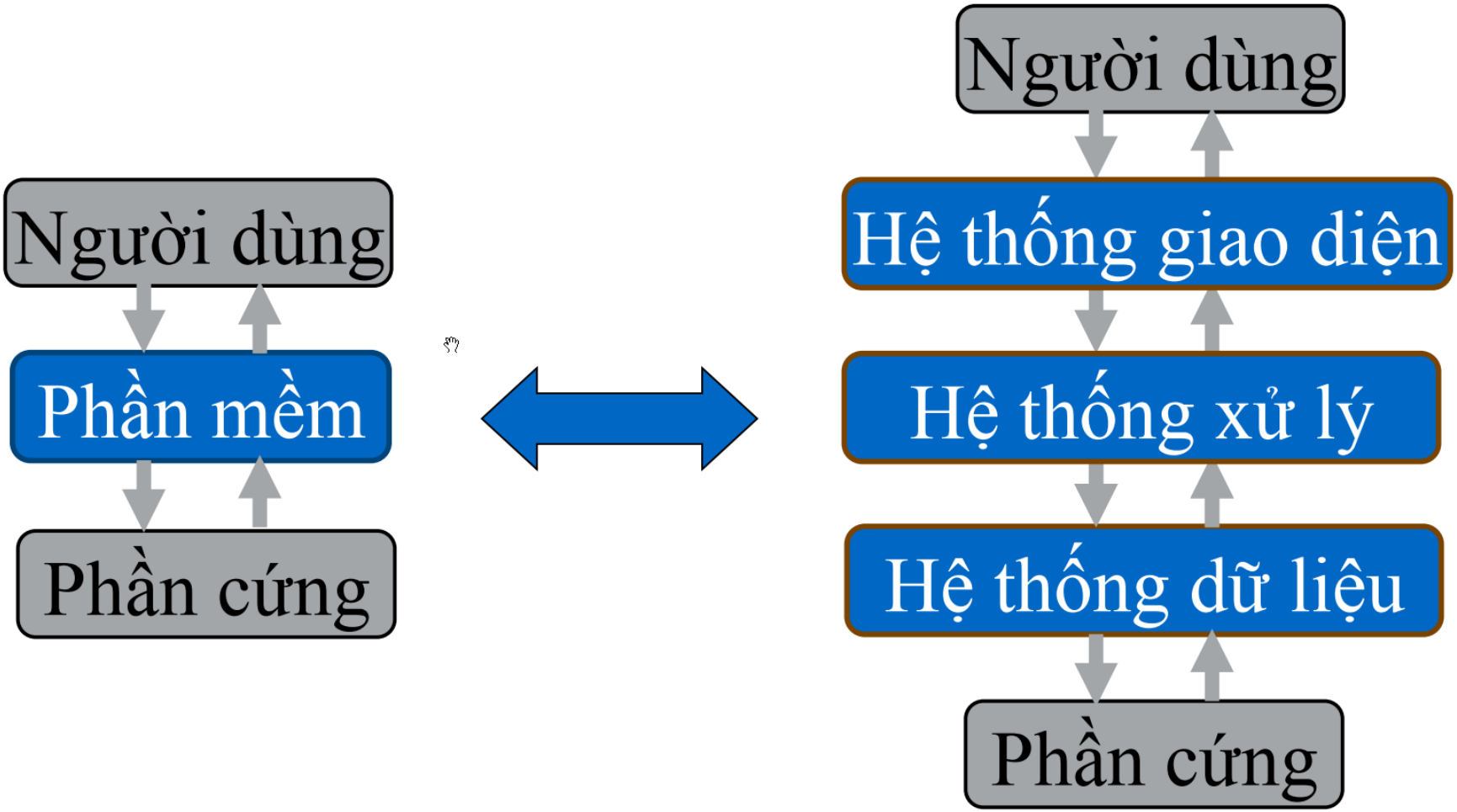
- Phân loại theo khả năng ứng dụng:
  - Phần mềm dùng chung (sản phẩm đại trà – Generic Product):
    - Tác giả sở hữu đặc tả
    - Bán rộng rãi ngoài thị trường
    - Đối tượng người sử dụng là tương đối đa dạng và phong phú
    - VD: Windows, Microsoft Office, Matlap, các game...
    - Ưu điểm: có khả năng ứng dụng rộng rãi cho nhiều nhóm người sử dụng
    - Nhược điểm: thiếu tính uyển chuyển, tùy biến

Phân loại phần mềm

- Phân loại theo khả năng ứng dụng:
  - Phần mềm đặt hàng (sản phẩm theo đơn đặt hàng - Bespoke Product hoặc Customised Product):
    - Tác giả sở hữu đặc tả
    - Làm riêng cho một khách hàng cụ thể theo yêu cầu
    - VD: Hệ thống quản lý đào tạo trường đại học, hệ thống quản lý thuế, hệ thống quản lý ngân hàng...
    - Ưu điểm: có tính uyển chuyển, tùy biến cao để đáp ứng được nhu cầu của một nhóm người sử dụng
    - Nhược điểm: ứng dụng trong chuyên ngành hẹp, chi phí cao

Phân loại phần mềm

Cấu trúc phần mềm



Tiêu chí phần mềm tốt

- 4 tiêu chí:
  - Tính bảo trì được (Maintainability)
    - Phần mềm phải được điều chỉnh và mở rộng, tiến hóa để đáp ứng các nhu cầu liên tục thay đổi
  - Tính tin cậy được (Reliability-Dependability)
    - Phần mềm phải được tin cậy, bảo mật và chính xác
  - Tính hiệu quả (Efficiency)
    - Phần mềm không nên sử dụng tài nguyên hệ thống một cách lãng phí
  - Tính chấp nhận được (Acceptability-Usability)
    - Phần mềm thỏa mãn được yêu cầu của người dùng: người dùng hiểu được, dùng được nó, và nó tương thích với các hệ thống khác

Tiêu chí phần mềm tốt

- Chất lượng phần mềm (dưới góc nhìn của người sử dụng):
  - Tính đúng đắn: Đầy đủ, Chính xác
  - Tính tiện dụng: Dễ học, Dễ sử dụng, Giao diện trực quan, Tự nhiên
  - Tính hiệu quả: Tối ưu sử dụng CPU, bộ nhớ, Tối ưu sử dụng thiết bị
  - Tính tương thích: Import/Export dữ liệu, Tương tác phần mềm khác
  - Tính tiến hóa

Tiêu chí phần mềm tốt

- Chất lượng phần mềm (dưới góc nhìn của chuyên viên CNTT):
  - Tính dễ kiểm tra: việc kiểm tra các thành phần phù hợp với yêu cầu phần mềm là dễ dàng nhất có thể được
  - Tính dễ sửa lỗi: khi có sự không phù hợp (so với yêu cầu) trong quá trình kiểm tra một thành phần, việc phát hiệu chính xác "vị trí lỗi" và sửa lỗi là nhanh nhất có thể được.
  - Tính dễ bảo trì: khi cần nâng cấp, cải tiến một thành phần (theo yêu cầu mới), việc cập nhật phần mềm là nhanh, chính xác nhất có thể được và cố gắng hạn chế ảnh hưởng đến các thành phần khác
  - Tính tái sử dụng: các thành phần đã thực hiện có thể dùng lại trong các phần mềm cùng lớp với thời gian và công sức ít nhất có thể được

Tiêu chí phần mềm tốt

Customer:
solves problems at
an acceptable cost in
terms of money paid and
resources used

User:
easy to learn;
efficient to use;
helps get work done

Developer:
easy to design;
easy to maintain;
easy to reuse its parts

Development manager:
sells more and
pleases customers
while costing less
to develop and maintain

1.2. Vai trò của phần mềm

- Phần mềm có vai trò quan trọng
  - Làm thay đổi phong cách làm việc của tổ chức
  - Tăng hiệu suất làm việc của đơn vị
  - Ảnh hưởng đến nền kinh tế quốc gia
    - Nền kinh tế của tất cả các nước phát triển đều phụ thuộc vào phần mềm
    - Càng ngày càng có nhiều hệ thống được kiểm soát bởi phần mềm
    - Tiền chi cho phần mềm chiếm một tỷ lệ quan trọng trong GDP của tất cả các nước phát triển
    - Do đó, việc xây dựng và bảo trì hệ thống phần mềm một cách hiệu quả là yêu cầu cần thiết đối với nền kinh tế toàn cầu và của từng quốc gia

1.3. Đặc trưng của phần mềm

- Phần mềm vốn phức tạp
- Yêu cầu phần mềm không ngừng thay đổi
  - Nhu cầu con người
  - Quy trình quản lý
  - Hạ tầng phần cứng
- Nhu cầu sử dụng phần mềm ngày càng tăng lên
- Phát triển phần mềm là công việc phức tạp, rủi ro

Các vấn đề về phần mềm

- Thoái hóa theo thời gian
  - Môi trường sử dụng
  - Nhu cầu thay đổi của người sử dụng
  - Lỗi sinh ra do nâng cấp
- Không có danh mục thay thế
  - Không có danh mục chi tiết sản phẩm thay thế cho trước
  - Sản phẩm đặt hàng theo từng yêu cầu riêng

Các vấn đề về phần mềm

- Tính phức tạp
  - Phần mềm là hệ thống logic lưu trên vật mang (giấy, thiết bị), khó hiểu, khó nắm bắt được
    - Nhiều khái niệm khác nhau, khó hiểu
    - Liên kết các khái niệm là lôgic
    - Để nắm, hiểu phải qua quá trình tư duy trừu tượng
  - Không nhìn thấy được
    - Không phải vật thể vật lý
    - Mỗi biểu diễn chỉ thể hiện một khía cạnh của hệ thống (dữ liệu, hành vi, cấu trúc, giao diện), không phải hệ thống tổng thể.

Các vấn đề về phần mềm

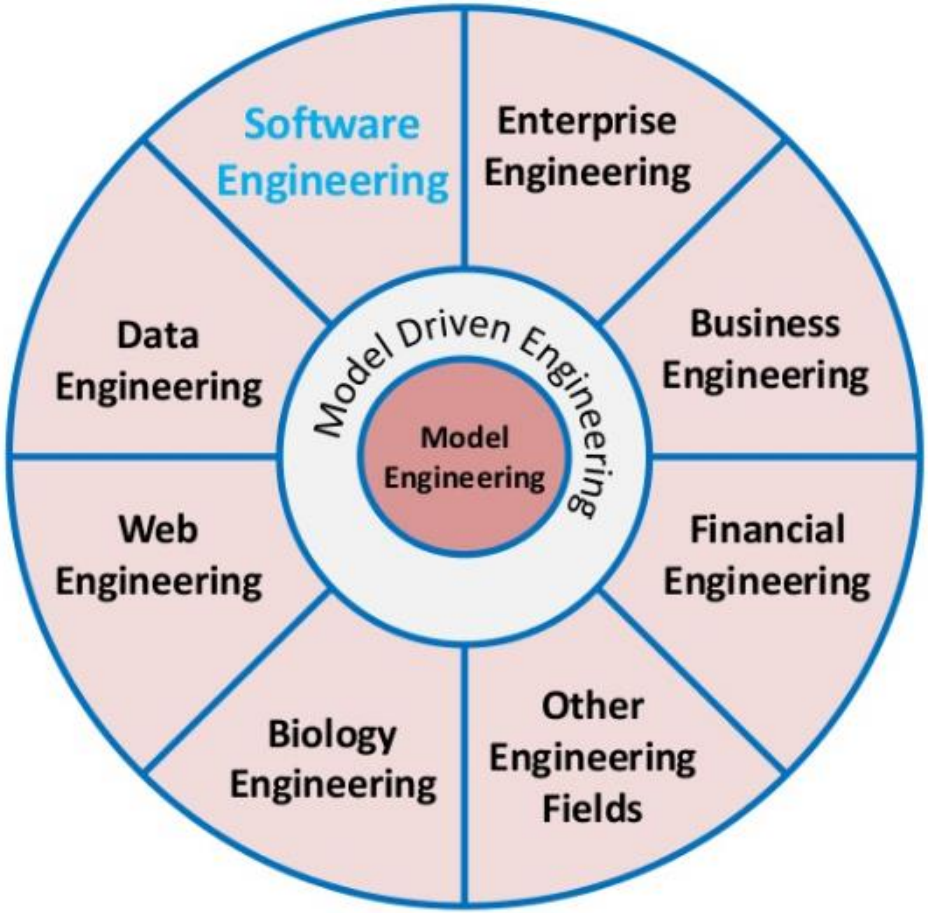
- Tính thay đổi
  - Phần mềm là mô hình thế giới thực, thay đổi theo thời gian
    - Mọi đối tượng thế giới thay đổi theo thời gian
    - Nhu cầu con người thay đổi theo thời gian
    - Phần mềm phải thích ứng với thế giới mà nó mô tả và yêu cầu của người dùng
- Phần mềm được phát triển theo nhóm
  - Năng suất của nhóm không tỷ lệ với số thành viên
  - Thời gian trao đổi thông tin chiếm tỷ lệ cao

Khái niệm công nghệ phần mềm

- Công nghệ phần mềm (Kỹ thuật/Kỹ nghệ phần mềm - Software Engineering):
  - Các lý thuyết, các phương pháp và các công cụ hỗ trợ cho phát triển phần mềm
    - Áp dụng các lý thuyết, các phương pháp, các công cụ phù hợp trong quá trình sản xuất phần mềm dưới các ràng buộc về tổ chức và tài chính.
    - Phát triển các lý thuyết, các phương pháp, các công cụ hỗ trợ quá trình sản xuất phần mềm.

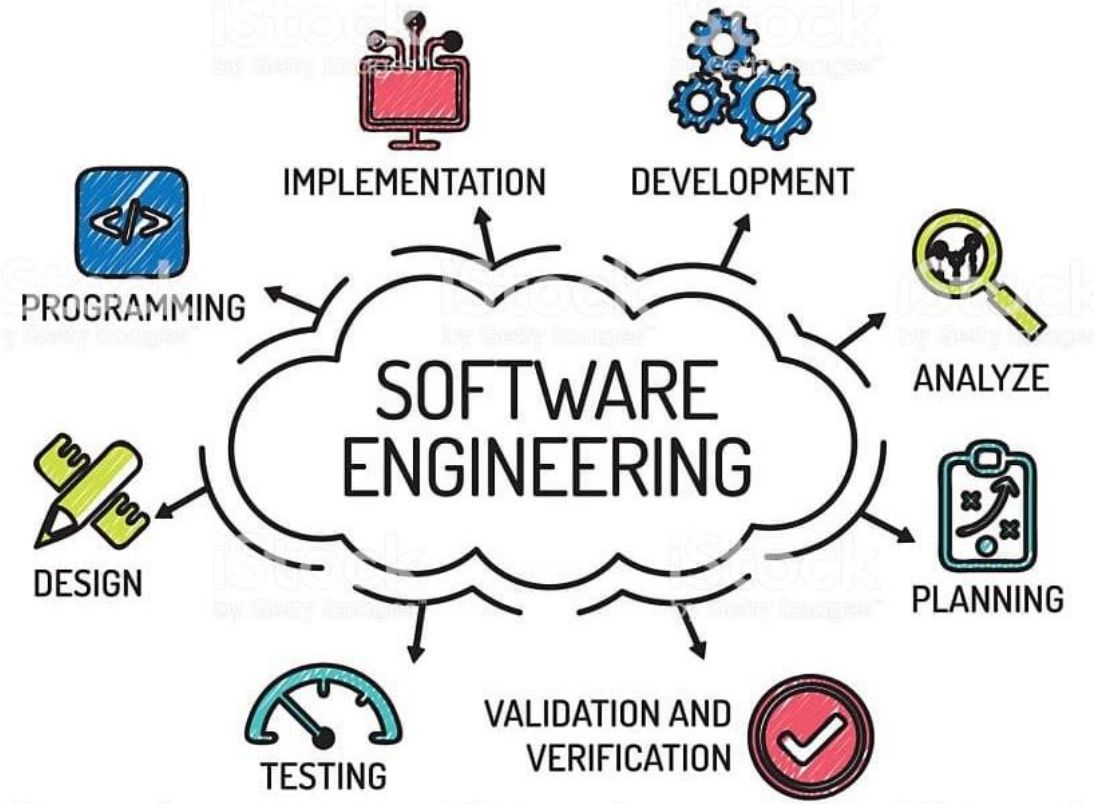
Khái niệm công nghệ phần mềm

- Công nghệ phần mềm (Kỹ thuật/Kỹ nghệ phần mềm - Software Engineering):

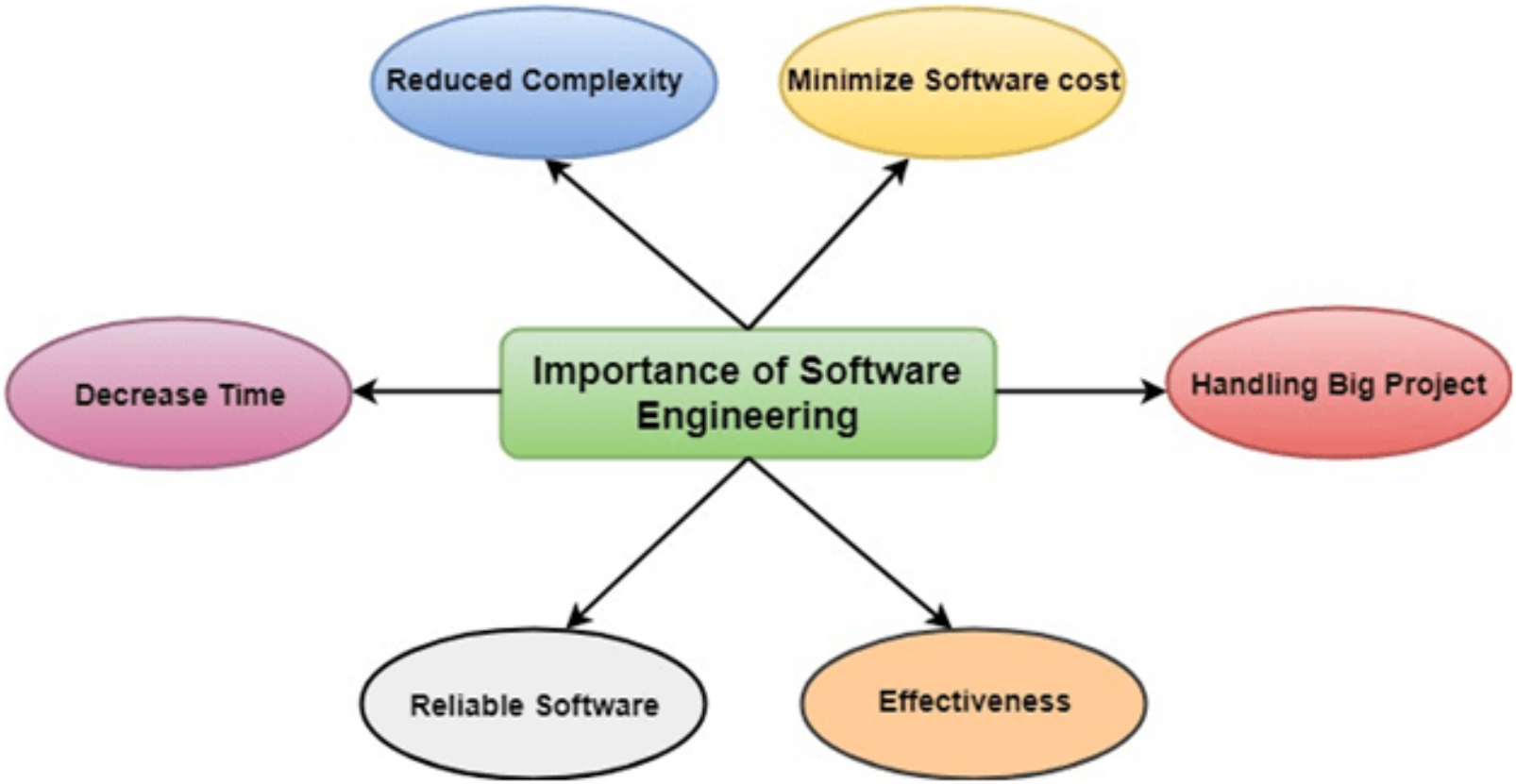


Khái niệm công nghệ phần mềm

- Công nghệ phần mềm (Kỹ thuật/Kỹ nghệ phần mềm - Software Engineering):



Sự quan trọng của Công nghệ phần mềm



Khái niệm công nghệ phần mềm

- Một số định nghĩa về Công nghệ phần mềm:
  - Bauer [1969]: SE là việc thiết lập và sử dụng các nguyên lý công nghệ đúng đắn để thu được phần mềm một cách kinh tế vừa tin cậy vừa làm việc hiệu quả trên các máy thực.
  - Sommerville [1995]: SE là một nguyên lý công nghệ liên quan đến tất cả các mặt (lý thuyết, phương pháp và công cụ) của sản phần mềm.
  - IEEE [1993]:
    - 1. Việc áp dụng phương pháp tiếp cận có hệ thống, bài bản và được lượng hóa trong phát triển, vận hành và bảo trì phần mềm.
    - 2. Nghiên cứu các phương pháp tiếp cận được dùng trong (1).
  - Pressman [1995]: SE là bộ môn tích hợp cả qui trình, các phương pháp, các công cụ để phát triển phần mềm máy tính.

Các yếu tố trong công nghệ phần mềm

- Phương pháp (method)
  - Cách làm cụ thể để xây dựng phần mềm
  - Mỗi công đoạn làm phần mềm có các phương pháp riêng
    - Phương pháp phân tích (xác định, đặc tả yêu cầu)
    - Phương pháp thiết kế (đặc tả thiết kế, thuật toán, dữ liêu)
    - Phương pháp lập trình (cấu trúc, hướng đối tượng)
    - Phương pháp kiểm thử (hộp đen, hộp trắng, luồn sợi)

Các yếu tố trong công nghệ phần mềm

- Quy trình (process)
  - Các bước thực hiện và thứ tự các bước; đầu vào, đầu ra ở mỗi bước
  - Xác định các tài liệu, sản phẩm cần bàn giao, và cách thức thực hiện
  - Định các mốc thời gian (millestones) và sản phẩm bàn giao (theo các chuẩn)
    - Có thể ở mức chung cho nhiều dự án
    - Hay cụ thể hoá cho 1 dự án cụ thể

Các yếu tố trong công nghệ phần mềm

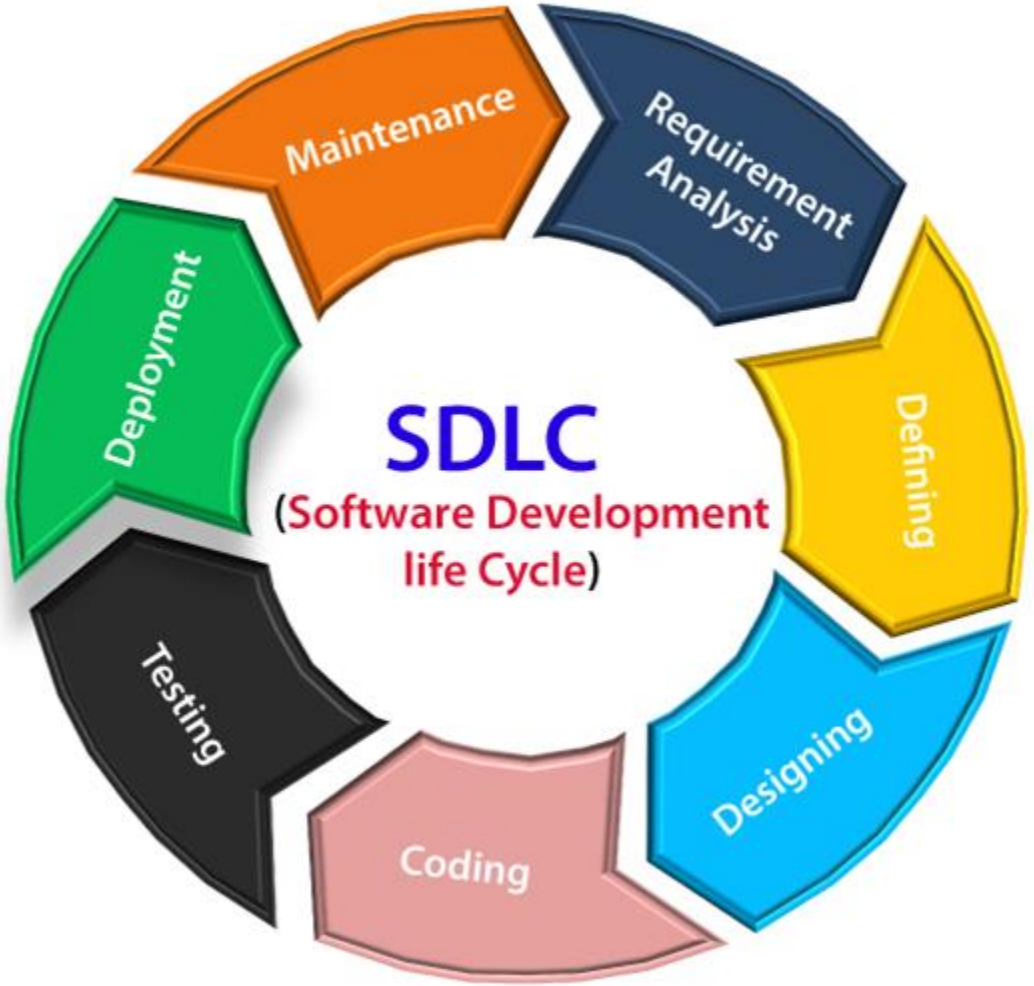
- Công cụ (tool)
  - Thực hiện tự động/bán tự động các công đoạn làm phần mềm
  - Các công cụ phần mềm trợ giúp các công đoạn khác nhau trong quá trình phát triển phần mềm (Computer Aided Software Engineering – CASE)
    - Hỗ trợ quản lý dự án: Microsoft Project...
    - Hỗ trợ phân tích, thiết kế: Power Designer...
    - Hỗ trợ viết chương trình, biên dịch, gỡ lỗi: Jbuilder...
    - Hỗ trợ kiểm thử: Defect Managerment System...
  - Các ngôn ngữ lập trình

📖 Vòng đời phát triển phần mềm

- Systems Development Life Cycle - SDLC
  - Là các hoạt động thực hiện tính từ khi phần mềm được đề xuất cho đến khi loại bỏ, cụ thể là từ khi được đặt hàng, phát triển, sử dụng đến khi bị loại bỏ.
  - Vòng đời phần mềm được phân chia thành các pha chính như xác định yêu cầu, triển khai, kiểm thử, bảo trì (vận hành). Phạm vi, thứ tự các pha khác nhau tùy theo từng sản phẩm và từng cách thức tiến hành.

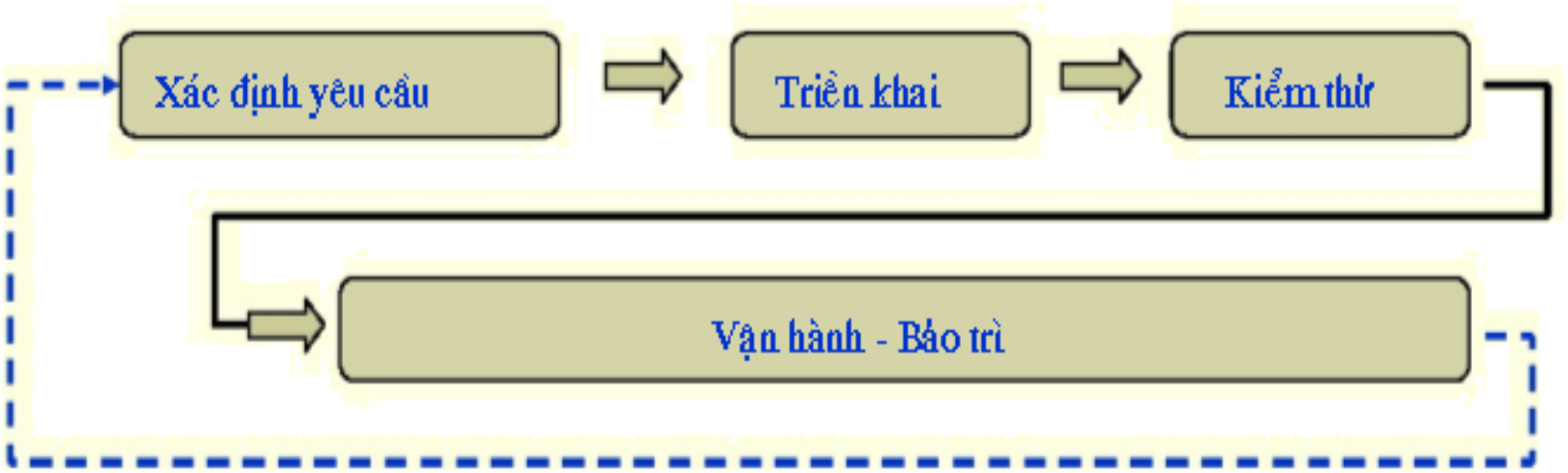
Vòng đời phát triển phần mềm

- Systems Development Life Cycle - SDLC



Vòng đời phát triển phần mềm

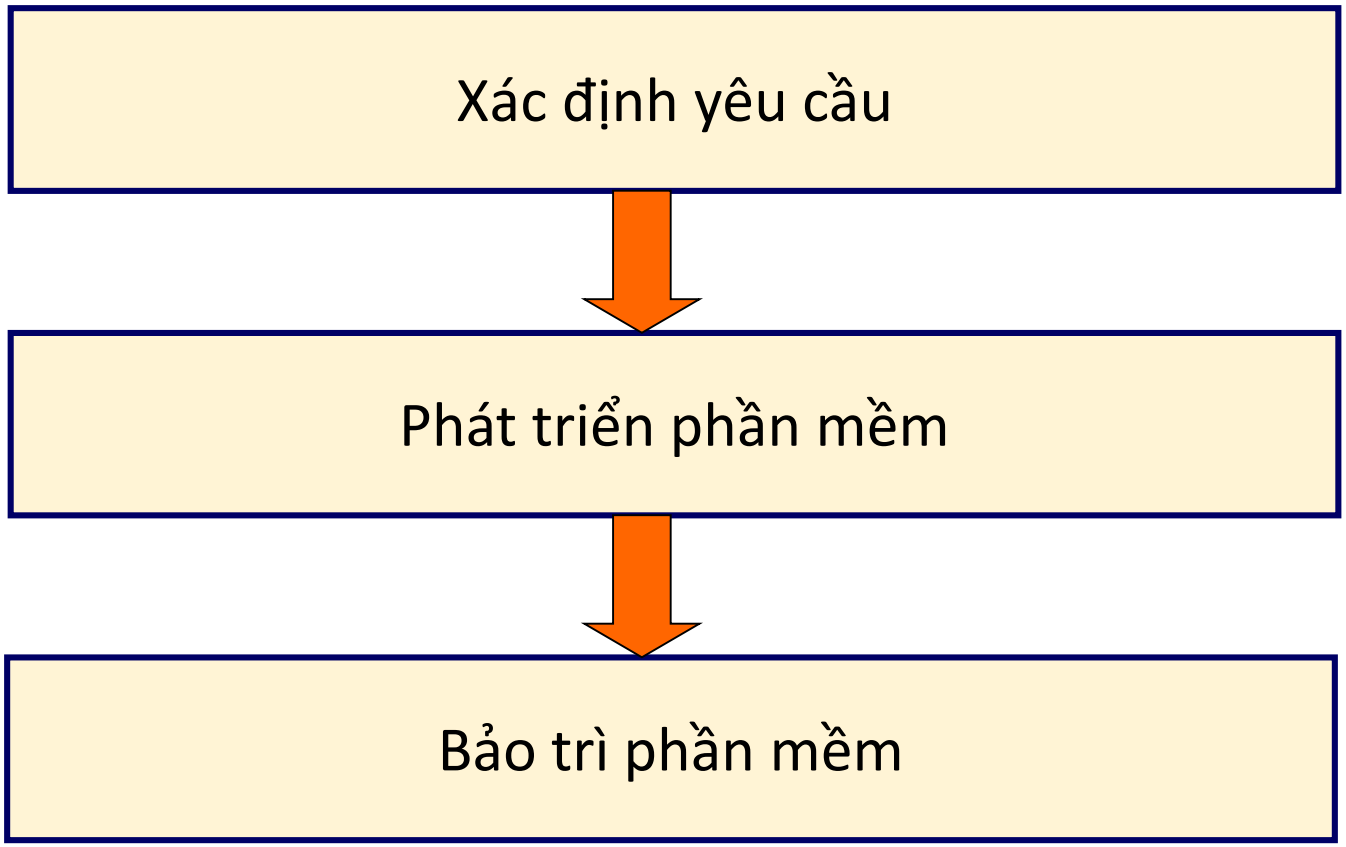
- Tùy mô hình áp dụng mà việc phân chia các pha (giai đoạn, từ 3 đến 20 bước).



Các bước lớn của vòng đời

Vòng đời phát triển phần mềm

- Các bước chung nhất phát triển phần mềm



📖 Vòng đời phát triển phần mềm

- Xác định yêu cầu: Xác định hệ thống làm gì? Những ràng buộc cần tuân thủ?
  - Phân tích hệ thống
    - Vai trò của phần mềm cần phát triển trong hệ thống
    - Phác hoạ và chọn một phương án phần mềm khả thi
  - Lập kế hoạch
    - Ước lượng công việc, lập lịch biểu, phân công công việc
  - Phân tích yêu cầu
    - Xác định yêu cầu chi tiết (chức năng, ràng buộc)
    - Đặc tả yêu cầu (kiến trúc, giao diện, xử lý, dữ liệu)

Vòng đời phát triển phần mềm

- Phát triển phần mềm: Tiến hành sản xuất phần mềm như thế nào?
  - Thiết kế (design)
    - Dịch các yêu cầu thành bản thiết kế (kiến trúc, dữ liệu, thủ tục xử lý, giao diện)
  - Mã hóa (coding)
    - Chuyển thiết kế thành chương trình máy tính (trong một ngôn ngữ lập trình)
  - Kiểm thử (testing)
    - Phát hiện và sửa lỗi chương trình (lỗi lập trình, lỗi thiết kế... kiểm thử đơn vị, kiểm thử tích hợp)

Vòng đời phát triển phần mềm

- Bảo trì phần mềm: Hoàn thiện hệ thống sau khi đưa vào hoạt động?
  - Sửa lỗi
    - Sửa lỗi phần mềm → vận hành thông suốt
  - Thích nghi
    - Sửa đổi để thích nghi với môi trường thay đổi, và để làm việc hiệu quả
  - Nâng cao
    - Thêm các chức năng mới, hoàn thiện chức năng cũ, phát triển dự phòng

Vòng đời phát triển phần mềm

- Mô hình tiến trình phần mềm (software process model)
  - Mô hình tiến trình phần mềm là một cách biểu diễn trừu tượng một tiến trình phần mềm đã đơn giản hóa và được thể hiện ra dưới một góc nhìn nào đó
  - Các góc nhìn tiến trình
    - Luồng công việc: Trình tự các hoạt động
    - Luồng dữ liệu: Luồng các dữ liệu di chuyển
    - Vai trò/hành động: Hành vi của tác nhân

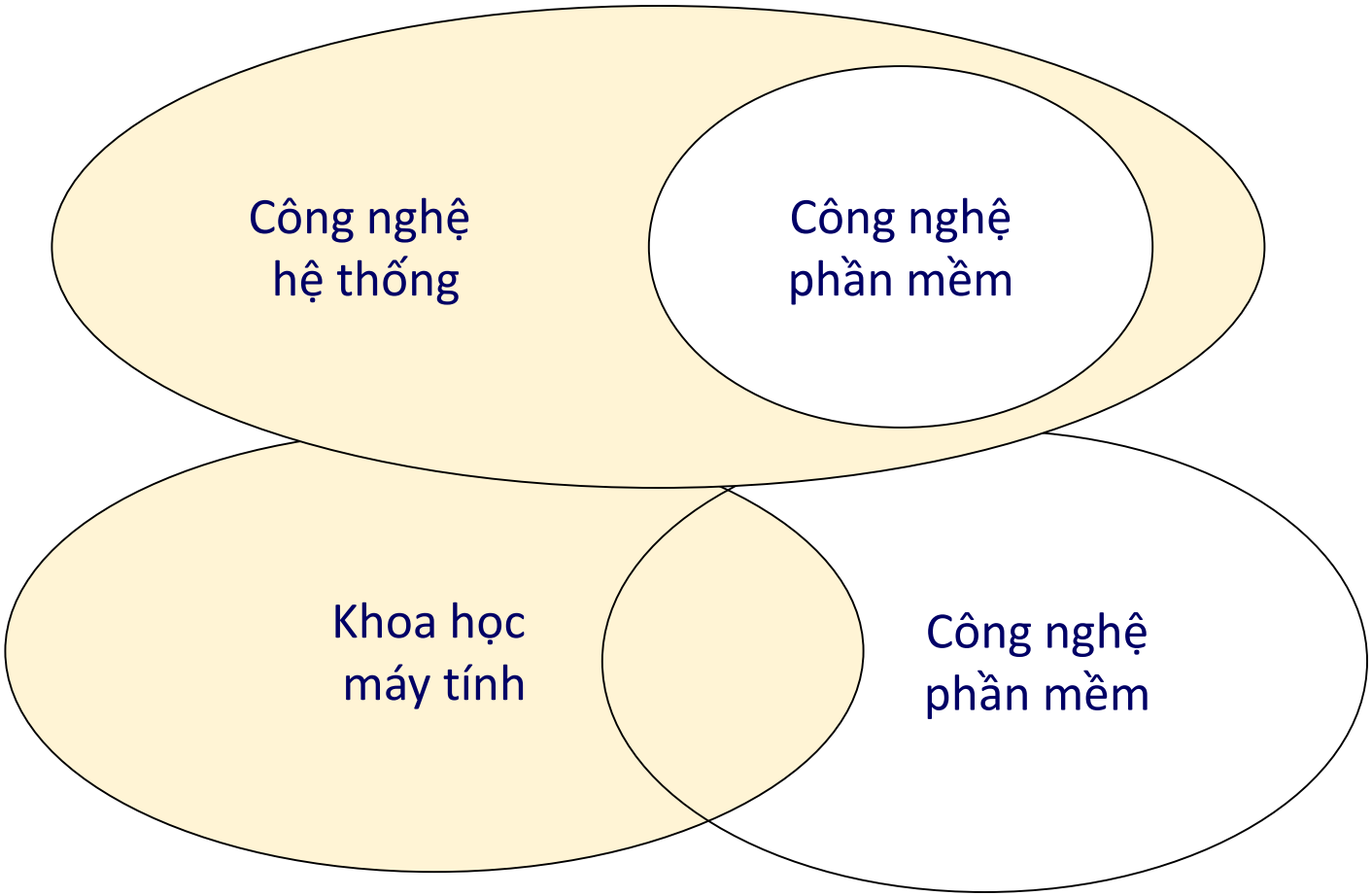
Các hoạt động chính

- Các hoạt động tổng quát trong mọi quy trình phần mềm:
  - Đặc tả (specification): hệ thống cần làm gì và các ràng buộc
  - Phát triển (development): tạo ra hệ thống phần mềm
  - Thẩm định/Kiểm thử (validation): kiểm tra xem phần mềm có đúng như khách hàng muốn hay không
  - Tiến hóa/Mở rộng (evolution): sửa đổi phần mềm để đáp ứng các nhu cầu thay đổi
- Các kỹ sư phần mềm nên tuân theo một phương pháp luận có hệ thống và có tổ chức trong công việc, đồng thời nên sử dụng các công cụ và kỹ thuật thích hợp với vấn đề cần giải quyết, các ràng buộc và tài nguyên sẵn có.

So sánh công nghệ phần mềm

- Sự khác biệt giữa công nghệ phần mềm và công nghệ hệ thống, khoa học máy tính?
  - Khoa học máy tính liên quan đến những vấn đề lý thuyết và nền tảng cho máy tính và phần mềm.
  - Công nghệ hệ thống (hay còn gọi là kỹ nghệ hệ thống) liên quan tới tất cả các khía cạnh của quá trình phát triển hệ thống dựa máy tính bao gồm: phần cứng, phần mềm, và công nghệ xử lý.
  - Công nghệ phần mềm chỉ là một phần của quy trình này, nó có liên quan tới việc phát triển hạ tầng phần mềm (software infrastructure), điều khiển, các ứng dụng và cơ sở dữ liệu trong hệ thống.
- Kỹ sư hệ thống phải thực hiện việc đặc tả hệ thống, thiết kế kiến trúc hệ thống, tích hợp và triển khai.

So sánh công nghệ phần mềm



Thách thức đối với công nghệ phần mềm

- Công nghệ phần mềm trong thế kỷ 21 phải đối mặt với rất nhiều thách thức to lớn. Với mỗi thách thức này, chúng ta phải có những giải pháp cụ thể. Các thách thức bao gồm:
  - Không đồng nhất:
    - Phát triển các kỹ thuật xây dựng phần mềm để giải quyết sự không đồng nhất về môi trường thực hiện và nền tảng hạ tầng.
  - Chuyển giao:
    - Phát triển các kỹ thuật nhằm dẫn tới việc chuyển giao phần mềm tới người sử dụng nhanh hơn.
  - Độ tin cậy:
    - Phát triển các kỹ thuật để chứng minh rằng phần mềm được người sử dụng tin tưởng.

Lịch sử phát triển

- Phần mềm phát triển không ngừng cùng sự tiến bộ của phần cứng
  - Về quy mô, sự phức tạp và tốc độ
  - Về chức năng. mức hoàn thiện
- Công nghệ không ngừng phát triển, cải tiến
- Khó khăn, thách thức ngày càng nhiều

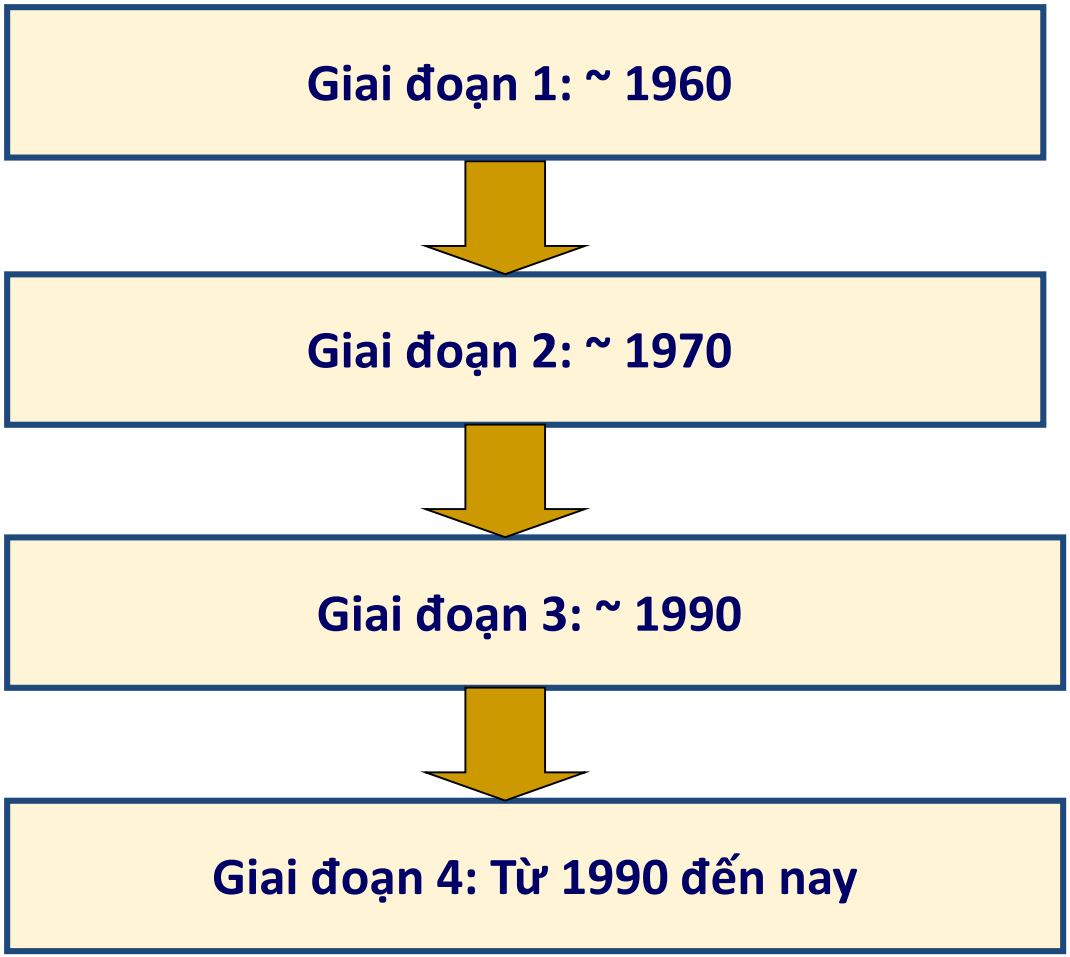
Lịch sử phát triển

- Năm 1968: Tại Tây Đức, hội nghị khoa học của NATO đã đưa ra từ "Software Engineering". Bắt đầu bàn luận về khủng khoảng phần mềm và xu hướng hình thành CNPM như một ngành riêng.
- Giữa những năm 1970: Hội nghị quốc tế đầu tiên về CNPM được tổ chức (1975): International Conference on SE (ICSE).
- Nửa sau những năm 1970: Cuộc "cách tân sản xuất phần mềm" đã bắt đầu trên phạm vi các nước công nghiệp.
- Nửa đầu những năm 1980: Xuất hiện các sản phẩm phần mềm và các công cụ khác nhau làm tăng năng suất sản xuất phần mềm đáng kể, phát triển các kỹ thuật bảo trì.

Lịch sử phát triển

- Nửa cuối những năm 1980 đến nay: Chất lượng phần mềm tập trung chủ yếu ở năng suất, độ tin cậy và việc bảo trì.
- Nghiên cứu hỗ trợ tự động hóa sản xuất phần mềm. Nhiều trung tâm, viện nghiên cứu CNPM ra đời. Các trường đưa vào giảng dạy SE.
- Hiện nay:
  - Công nghiệp hóa sản xuất phần mềm bằng cách đưa những kỹ thuật công nghệ học (Engineering techniques) thành cơ sở khoa học của CNPM.
  - Vận dụng những lý luận trong sản xuất phần mềm và áp dụng các phương pháp luận một cách nhất quán.
  - Tăng cường nghiên cứu và tạo công cụ trợ giúp sản xuất phần mềm

Lịch sử phát triển



🕮 Lịch sử phát triển

- Giai đoạn 1: ~ 1960
  - Phần cứng yếu, thay đổi liên tục, có tính chuyên dụng
  - Xử lí theo lô
  - Môi trường lập trình có tính cá nhân
  - Sản xuất đơn chiếc, chuyên dụng
  - Ngôn ngữ lập trình: mã máy, đặc thù cho từng máy
  - Phương pháp lập trình: là một nghệ thuật, chưa có phương pháp (cách làm: thử - sửa sai)

Lịch sử phát triển

- Giai đoạn 2: ~ 1970
  - Phần mềm trở thành sản phẩm, phân phối rộng rãi
  - Hệ thống đa nhiệm, đa người sử dụng, thời gian thực ra đời
  - Yêu cầu bảo trì nảy sinh
    - Nhu cầu sử dụng tăng cao
    - Bảo trì trở nên tốn kém, nhiều dự án thất bại
  - Ngôn ngữ lập trình: Có cấu trúc (Algol 60, Fortran, COBOL)
  - Phương pháp lập trình: Có phương pháp lập trình
  - Phát triển hệ thống: Chưa có phương pháp, kinh nghiệm là chính

Lịch sử phát triển

- Giai đoạn 3: ~ 1990
  - Bộ vi xử lí phát triển kéo theo sự phát triển của máy cá nhân, máy trạm
  - Mạng cục bộ, toàn cầu phát triển
  - Hệ thống phân tán, nhu cầu truy nhập dữ liệu
    - Phần mềm ngày càng phức tạp, ra đời các chuẩn
    - Ra đời phần mềm nhúng
    - Số người dùng tăng nhanh
  - Ngôn ngữ lập trình: Bậc cao, hướng đối tượng
  - Phương pháp lập trình: Có phương pháp lập trình
  - Phát triển hệ thống: Phương pháp cấu trúc hoàn thiện, nhiều công cụ tự động.

Lịch sử phát triển

- Giai đoạn 4: Từ 1990 đến nay
  - Kỹ nghệ hướng đối tượng bắt đầu phát triển
  - Mạng toàn cầu phát triển nhanh, internet mở rộng
  - Hệ chuyên gia, trí tuệ nhân tạo, phần mềm nhúng đưa vào sử dụng rộng rãi
    - Năng lực máy tính đạt đến mức cao
    - Ra đời kỹ thuật "thế hệ thứ 4" phát triển phần mềm
    - Số người dùng tăng chóng mặt
  - Ngôn ngữ lập trình: Hướng đối tượng, thế hệ thứ 4 (.NET; Java...)
  - Phát triển hệ thống: Quy trình mới, tự động hoá cao (ROSE), sử dụng lại phổ biến (mẫu, mã nguồn mở...)

1.4. Các tiến trình phần mềm

- Tiến trình/quy trình phần mềm (software process):
  - Gồm một tập hợp các hoạt động được tổ chức mà mục đích của nó là xây dựng và phát triển phần mềm.
  - Quy trình phần mềm xác định một bộ khung và tiêu chuẩn để triển khai công nghệ phần mềm.
  - Tất cả các qui trình đều có những bước cơ sở: phân tích; thiết kế; hiện thực mã; kiểm thử; triển khai.
- Những loại hệ thống khác nhau sẽ cần những quy trình phát triển khác nhau.

1.4. Các tiến trình phần mềm

- Phân tích
  - Phân tích viên phân tích hiện trạng và yêu cầu của khách hàng
  - Mô hình hệ thống
  - Lập bảng các chức năng của hệ thống
  - Yêu cầu chức năng
  - Yêu cầu phi chức năng
  - Xác định phạm vi của hệ thống
- Thiết kế
  - Thiết kế dữ liệu, xử lý, giao diện
  - Thiết kế kiến trúc
  - Thiết kế thành phần…

1.4. Các tiến trình phần mềm

- Hiện thực mã: Tạo mã theo thiết kế
- Kiểm thử:
  - Kiểm thử đơn vị
  - Kiểm thử tích hợp
  - Kiểm thử thẩm tra hay kiểm thử tính năng
  - Kiểm thử hệ thống
- Triển khai hệ thống:
  - Giải pháp phần cứng
  - Cài đặt phần mềm
  - Chuyển giao công nghệ
  - Hướng dẫn sử dụng: đào tạo, tài liệu hướng dẫn
- Bảo trì và tiến hóa

📖 Phương pháp phát triển phần mềm

- Phương pháp hướng chức năng:
  - Xây dựng phần mềm dựa trên các chức năng mà hệ thống cần thực hiện.
  - Phương pháp chung để giải quyết vấn đề là áp dụng nguyên lý “chia để trị”.
  - Hạn chế: có khả năng các chức năng trong hệ thống không tương thích với nhau khi thực hiện thay đổi các thông tin trong hệ thống.

Phương pháp phát triển phần mềm

- Phương pháp hướng dữ liệu:
  - Chú trọng đến thành phần dữ liệu của hệ thống.
  - Dùng mô hình thực thể kết hợp để biểu diễn các thực thể và mối liên hệ giữa các thực thể.
  - Hạn chế: phần mềm chỉ có chức năng chính là lưu trữ và thao tác trên các đối tượng dữ liệu, không quan tâm đến các chức năng khác của hệ thống nên hệ thống thu được sau khi thiết kế có thể thiếu một số chức năng cần thiết.

🕮 Phương pháp phát triển phần mềm

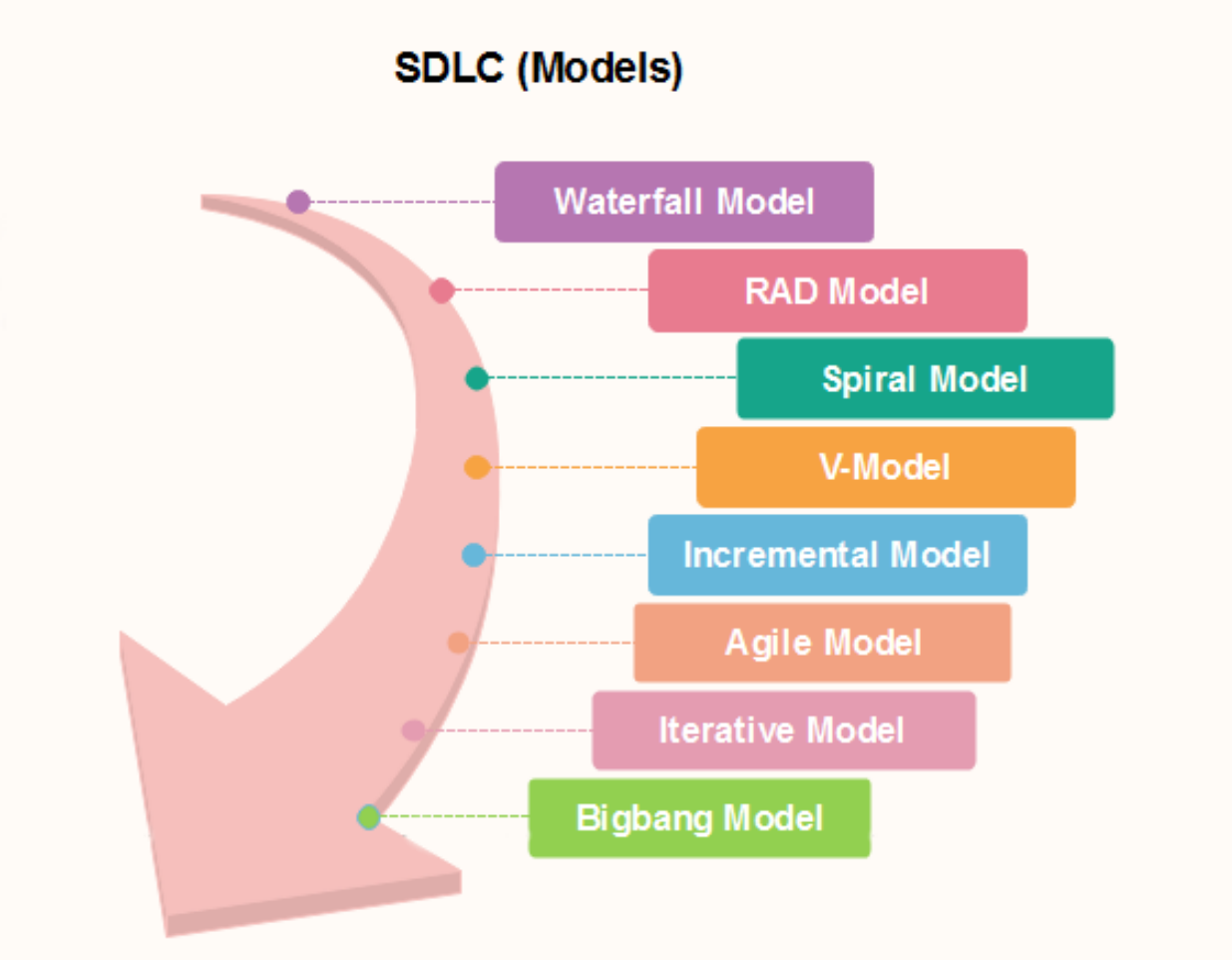
- Phương pháp hướng đối tượng:
  - Chú trọng đến thành phần dữ liệu và chức năng của hệ thống.
  - Hệ thống phần mềm là một tập hợp các đối tượng có khả năng tương tác với nhau.
  - Mỗi đối tượng bao gồm dữ liệu và các thao tác thực hiện trên dữ liệu của đối tượng.

1.4. Các tiến trình phần mềm

- Một số mô hình tiến trình/phát triển phần mềm (software process model):
  - Mô hình thác nước (waterfall)
  - Mô hình xoắn ốc (Spiral model)
  - Mô hình chữ V (V model)
  - Mô hình phát triển lặp (Iterative development)
  - Mô hình tăng trưởng (Incremental model)
  - Mô hình RAD (Rapid Application Development)
  - Mô hình Agile
  - Mô hình Scrum
  - Mô hình Big Bang

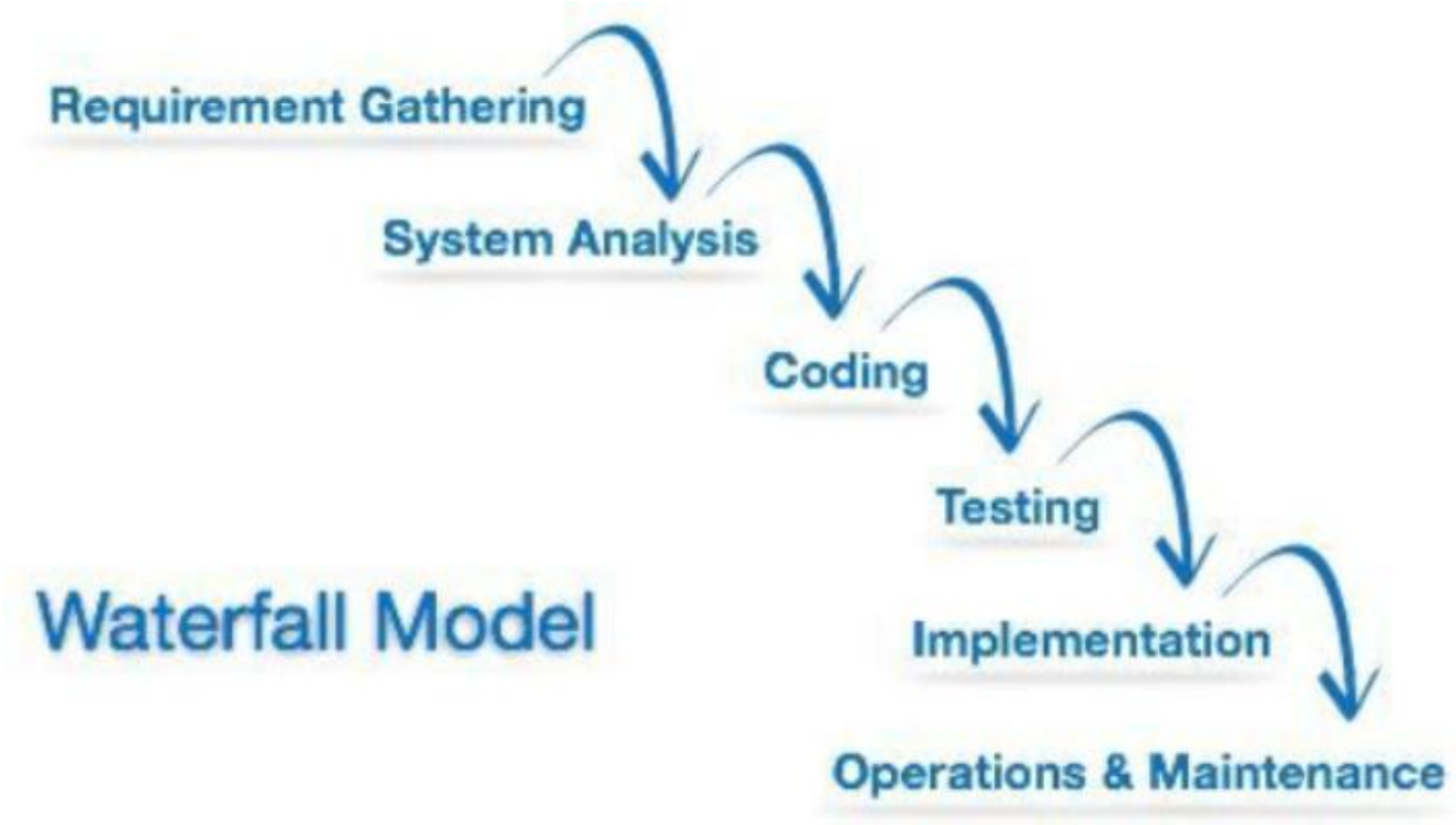
1.4. Các tiến trình phần mềm

- Một số mô hình tiến trình/phát triển phần mềm (software process model):



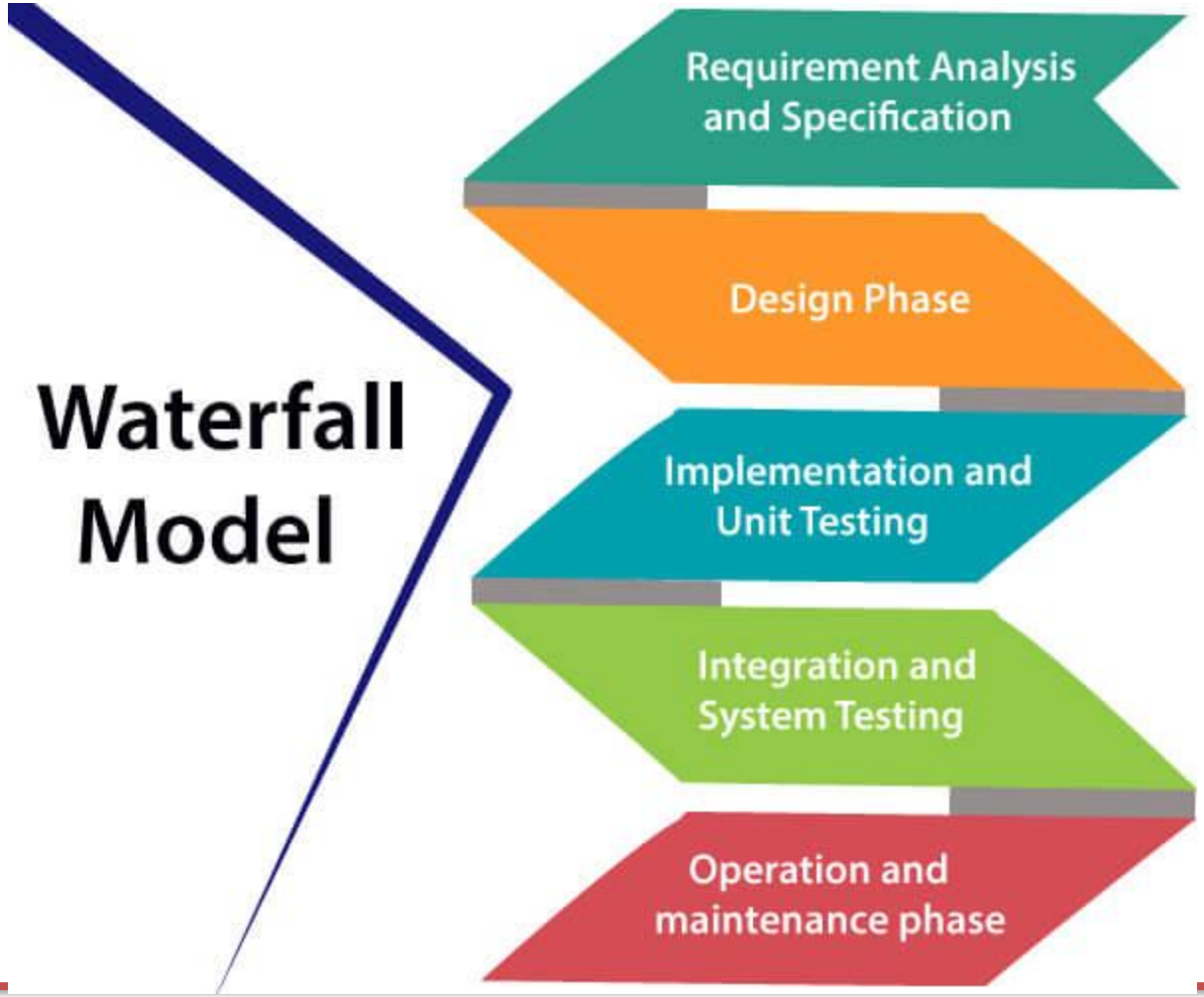
Tiến trình phần mềm

- Mô hình thác nước (waterfall)



Tiến trình phần mềm

- Mô hình thác nước (waterfall)



Tiến trình phần mềm

- Mô hình thác nước (Waterfall)
  - Mô tả
    - Đây được coi như là mô hình phát triển phần mềm đầu tiên được sử dụng.
    - Mô hình này áp dụng tuần tự các giai đoạn của phát triển phần mềm.
    - Đầu ra của giai đoạn trước là đầu vào của giai đoạn sau. Giai đoạn sau chỉ được thực hiện khi giai đoạn trước đã kết thúc.

Tiến trình phần mềm

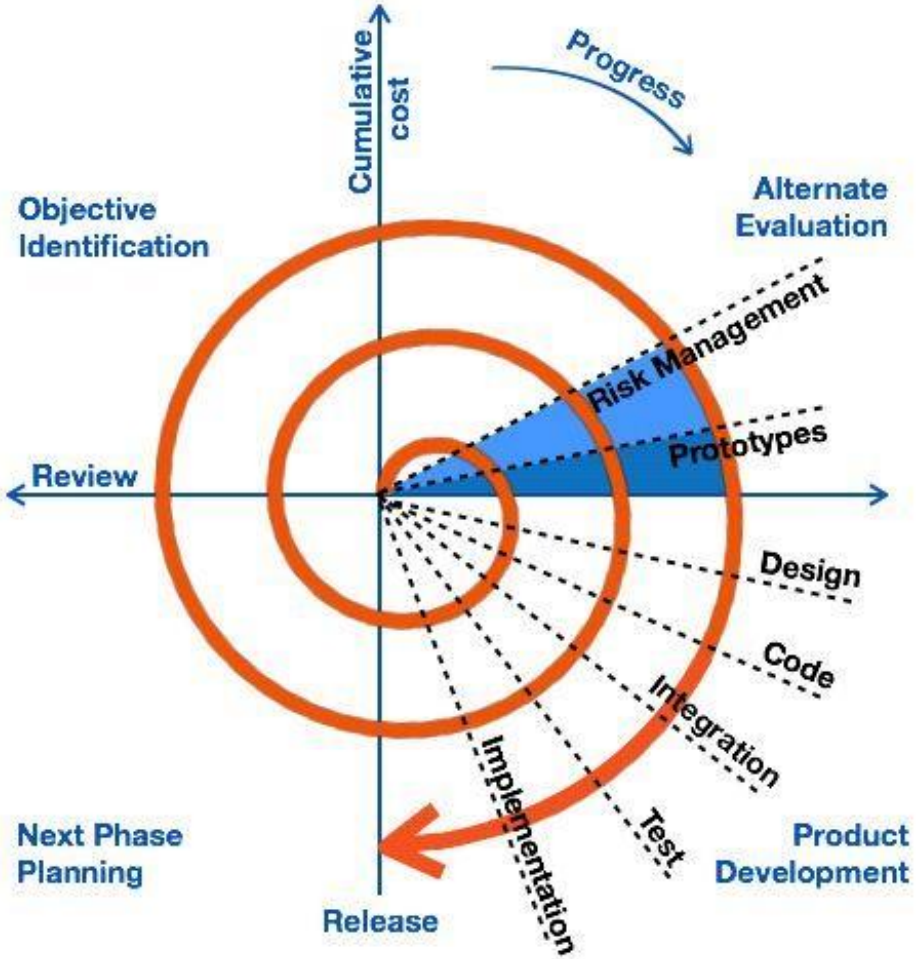
- Mô hình thác nước (waterfall)
  - Phân tích mô hình
    - Requirement gathering: Thu thập và phân tích yêu cầu được ghi lại vào tài liệu đặc tả yêu cầu trong giai đoạn này.
    - System Analysis: Phân tích thiết kế hệ thống phần mềm, xác định kiến trúc hệ thống tổng thể của phần mềm.
    - Coding: Hệ thống được phát triển theo từng unit và được tích hợp trong giai đoạn tiếp theo. Mỗi Unit được phát triển và kiểm thử bởi dev được gọi là Unit Test.
    - Testing: Cài đặt và kiểm thử phần mềm. Công việc chính của giai đoạn này là kiểm tra và sửa tất cả những lỗi tìm được sao cho phần mềm hoạt động chính xác và đúng theo tài liệu đặc tả yêu cầu.
    - Implementation: Triển khai hệ thống trong môi trường khách hàng và đưa ra thị trường.
    - Operations and Maintenance: Bảo trì hệ thống khi có bất kỳ thay đổi nào từ phía khách hàng, người sử dụng.

Tiến trình phần mềm

- Mô hình thác nước (waterfall)
  - Ứng dụng
    - Các dự án nhỏ, ngắn hạn.
    - Các dự án có ít thay đổi về yêu cầu và không có những yêu cầu không rõ ràng.
  - Ưu điểm
    - Dễ sử dụng, dễ tiếp cận, dễ quản lý.
    - Sản phẩm phát triển theo các giai đoạn được xác định rõ ràng.
    - Xác nhận ở từng giai đoạn, đảm bảo phát hiện sớm các lỗi.
  - Nhược điểm
    - Ít linh hoạt, phạm vi điều chỉnh hạn chế.
    - Rất khó để đo lường sự phát triển trong từng giai đoạn.
    - Mô hình không thích hợp với những dự án dài, đang diễn ra, hay những dự án phức tạp, có nhiều thay đổi về yêu cầu trong vòng đời phát triển.
    - Khó quay lại khi giai đoạn nào đó đã kết thúc.

Tiến trình phần mềm

- Mô hình xoắn ốc (Spiral model)



Tiến trình phần mềm

- Mô hình xoắn ốc (Spiral model)
  - Mô tả
    - Là mô hình kết hợp giữa các tính năng của mô hình prototyping và mô hình thác nước.
    - Mô hình xoắn ốc được ưa chuộng cho các dự án lớn, đắt tiền và phức tạp.
    - Mô hình này sử dụng những giai đoạn tương tự như mô hình thác nước, về thứ tự, plan, đánh giá rủi ro…

Tiến trình phần mềm

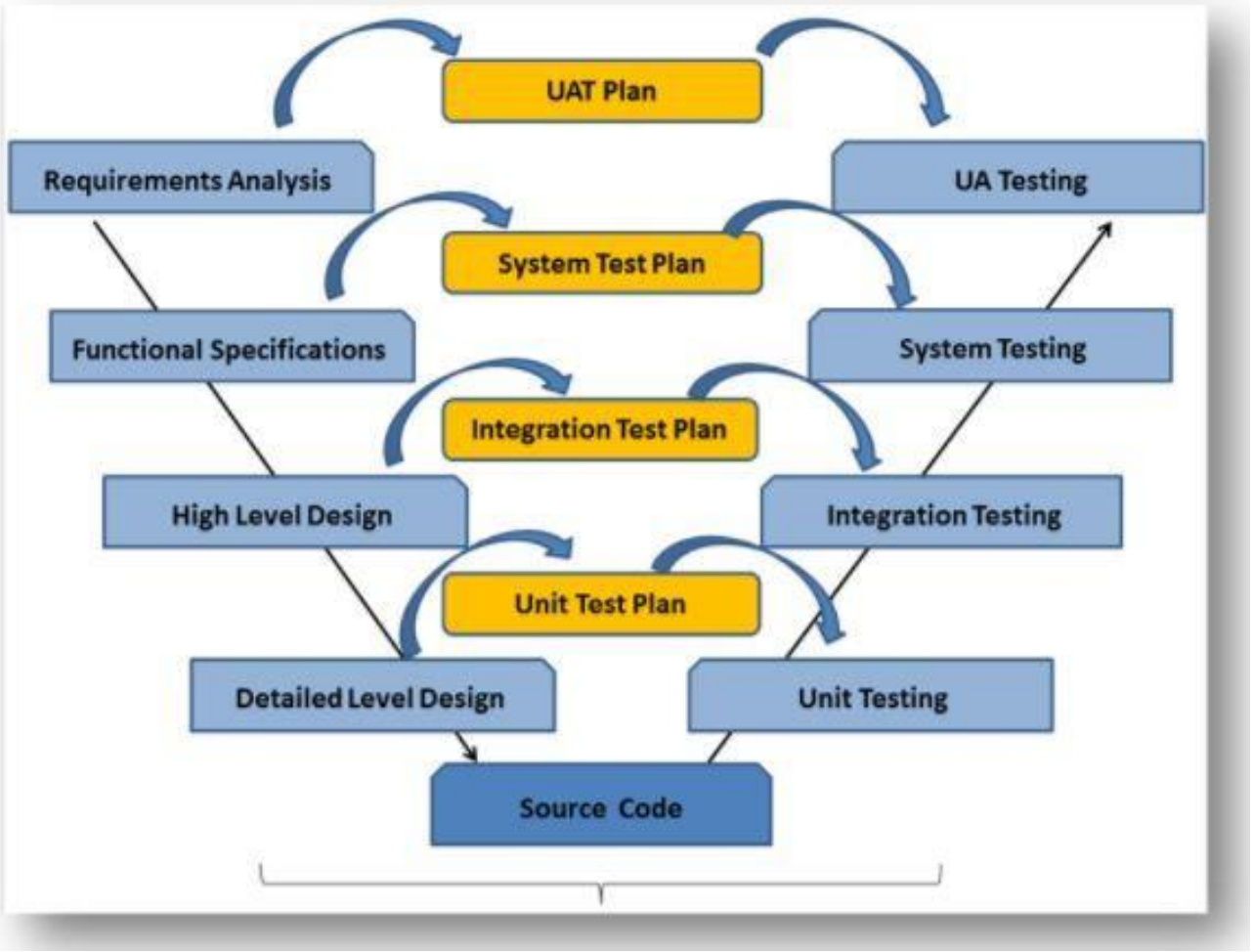
- Mô hình xoắn ốc (Spiral model)
  - Phân tích mô hình: Các pha trong quy trình phát triển xoắn ốc bao gồm:
    - Objective identification: Thiết lập mục tiêu: xác định mục tiêu, đối tượng cho từng pha của dự án.
    - Alternate evaluation: Đánh giá và giảm thiểu rủi ro: đánh giá rủi ro và thực hiện các hành động để giảm thiểu rủi ro.
    - Product development: Phát triển sản phẩm: Lựa chọn mô hình phù hợp để phát triển hệ thống.
    - Next phase planning: Lập kế hoạch: đánh giá dự án và lập kế hoạch cho pha tiếp theo.

Tiến trình phần mềm

- Mô hình xoắn ốc (Spiral model)
  - Ứng dụng
    - Sử dụng cho các ứng dụng lớn và các hệ thống được xây dựng theo các giai đoạn nhỏ hoặc theo các phân đoạn.
  - Ưu điểm
    - Tốt cho các hệ thống phần mềm quy mô lớn.
    - Dễ kiểm soát các mạo hiểm ở từng mức tiến hóa.
    - Đánh giá thực tế hơn như là một quy trình làm việc, bởi vì những vấn đề quan trọng đã được phát hiện sớm hơn.
  - Nhược điểm
    - Manager cần có kỹ năng tốt để quản lý dự án, đánh giá rủi ro kịp thời.
    - Chi phí cao và mất nhiều thời gian để hoàn thành dự án.
    - Phức tạp và không thích hợp với các dự án nhỏ và ít rủi ro.
    - Yêu cầu thay đổi thường xuyên dẫn đến lặp vô hạn.
    - Chưa được dùng rộng rãi.

Tiến trình phần mềm

- Mô hình chữ V (V model)



Tiến trình phần mềm

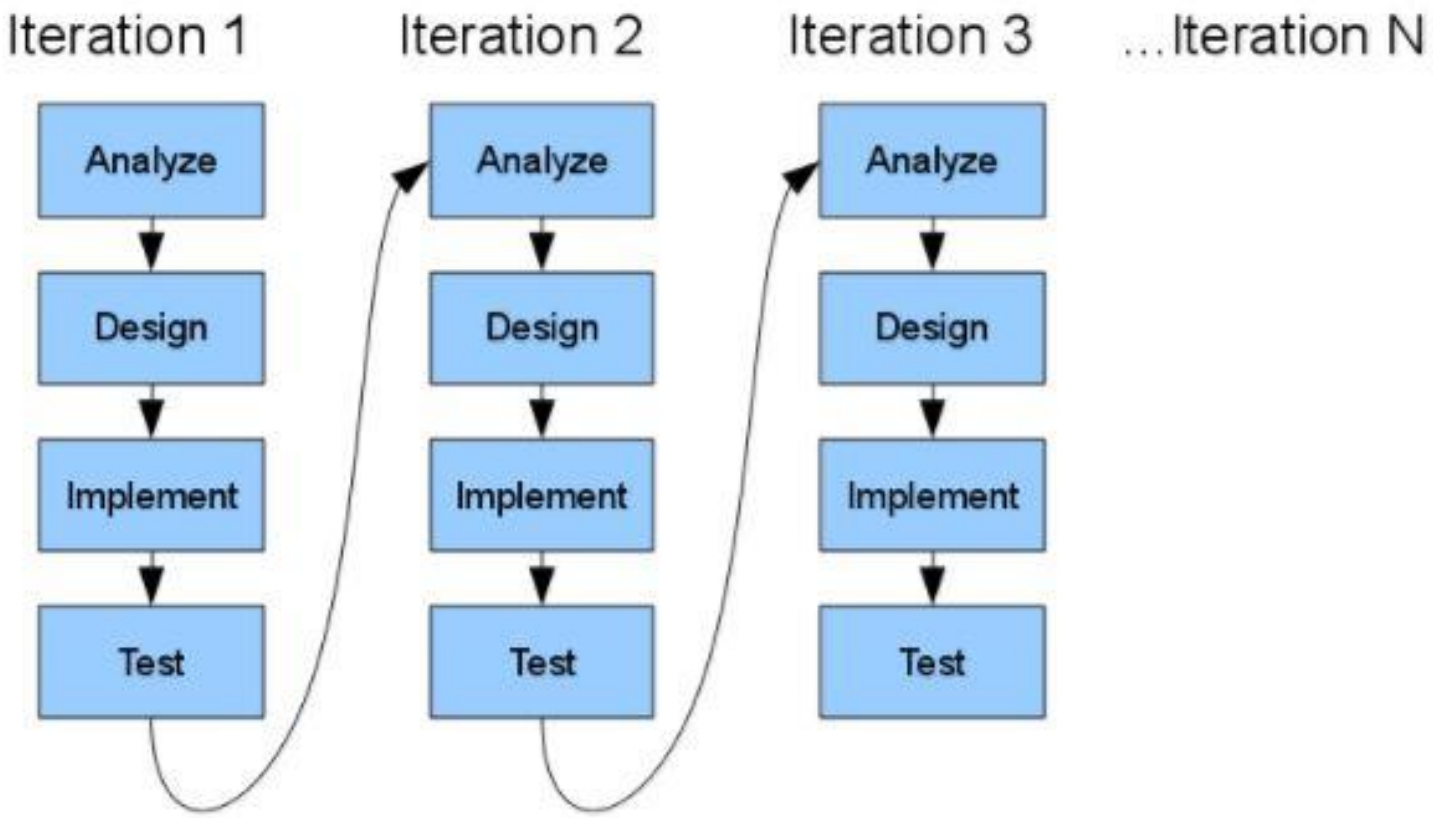
- Mô hình chữ V (V model)
  - Mô tả
    - Mô hình chữ V là một phần mở rộng của mô hình thác nước và được dựa trên sự kết hợp của một giai đoạn thử nghiệm cho từng giai đoạn phát triển tương ứng.
    - Đây là một mô hình có tính kỷ luật cao và giai đoạn tiếp theo chỉ bắt đầu sau khi hoàn thành giai đoạn trước.
    - Với V model thì công việc test được tham gia ngay từ đầu.
  - Ứng dụng
    - Yêu cầu được xác định rõ ràng.
    - Xác định sản phẩm ổn định.
    - Công nghệ không thay đổi và được hiểu rõ bởi nhóm dự án.
    - Không có yêu cầu không rõ ràng hoặc không xác định.
    - Dự án ngắn.

Tiến trình phần mềm

- Mô hình chữ V (V model)
  - Ưu điểm
    - Hoạt động tốt cho các dự án nhỏ, khi các yêu cầu được hiểu rất rõ.
    - Đơn giản và dễ hiểu và dễ sử dụng, dễ quản lý.
  - Nhược điểm
    - Khó quản lý kiểm soát rủi ro, rủi ro cao.
    - Không phải là một mô hình tốt cho các dự án phức tạp và hướng đối tượng.
    - Mô hình kém cho các dự án dài và đang diễn ra.
    - Không thích hợp cho các dự án có nguy cơ thay đổi yêu cầu trung bình đến cao.

Tiến trình phần mềm

- Mô hình phát triển lặp (Iterative development)



Tiến trình phần mềm

- Mô hình phát triển lặp (Iterative development)

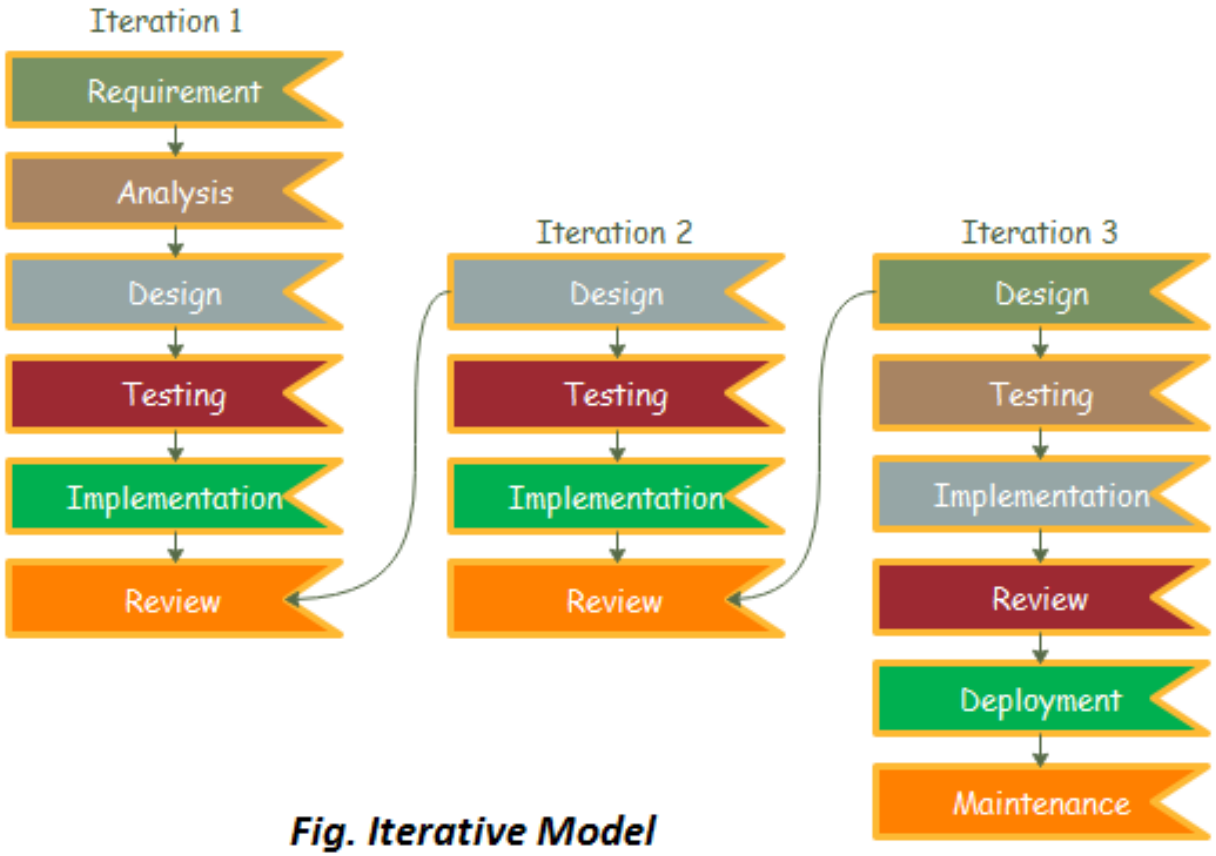


Fig. Iterative Model

Tiến trình phần mềm

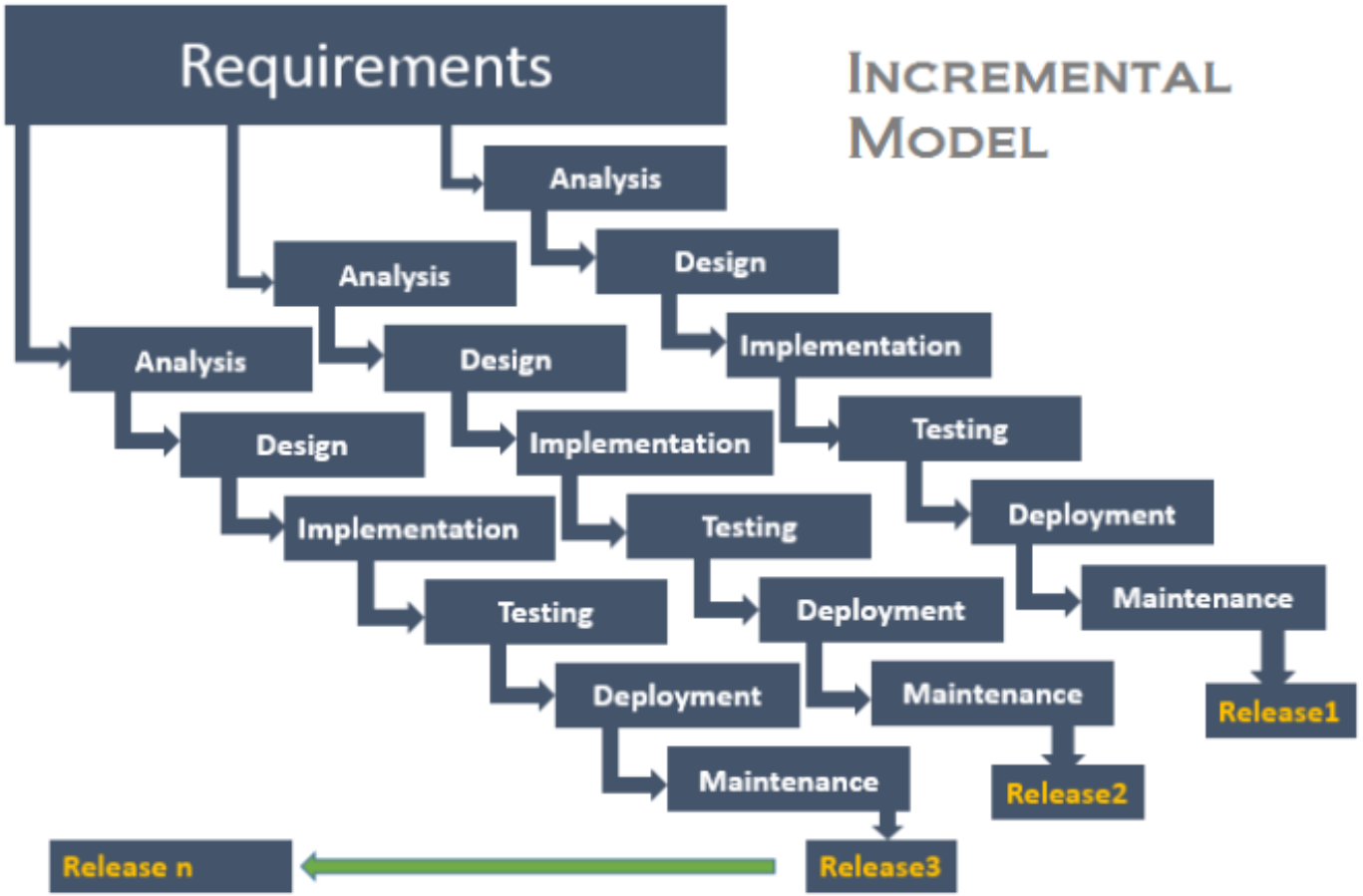
- Mô hình phát triển lặp (Iterative development)
  - Mô tả
    - Một mô hình được lặp đi lặp lại từ khi bắt đầu cho đến khi làm đầy đủ. Quá trình này sau đó được lặp lại, tạo ra một phiên bản mới của phần mềm vào cuối mỗi lần lặp của mô hình.
    - Thay vì phát triển phần mềm từ đặc tả rồi mới bắt đầu thực thi thì mô hình này có thể review dần dần để đi đến yêu cầu cuối cùng.
  - Ứng dụng
    - Yêu cầu chính phải được xác định; tuy nhiên, một số chức năng hoặc yêu cầu cải tiến có thể phát triển theo thời gian.
    - Một công nghệ mới đang được sử dụng và đang được học tập bởi nhóm phát triển trong khi làm việc trong dự án.
    - Phù hợp cho các dự án lớn và nhiệm vụ quan trọng.

Tiến trình phần mềm

- Mô hình phát triển lặp (Iterative development)
  - Ưu điểm
    - Xây dựng và hoàn thiện các bước sản phẩm theo từng bước.
    - Thời gian làm tài liệu sẽ ít hơn so với thời gian thiết kế.
    - Một số chức năng làm việc có thể được phát triển nhanh chóng và sớm trong vòng đời.
    - Ít tốn kém hơn khi thay đổ phạm vi, yêu cầu.
    - Dễ quản lý rủi ro.
    - Trong suốt vòng đời, phần mềm được sản xuất sớm để tạo điều kiện cho khách hàng đánh giá và phản hồi.
  - Nhược điểm
    - Yêu cầu tài nguyên nhiều.
    - Các vấn đề về thiết kế hoặc kiến trúc hệ thống có thể phát sinh bất cứ lúc nào.
    - Yêu cầu quản lý phức tạp hơn.
    - Tiến độ của dự án phụ thuộc nhiều vào giai đoạn phân tích rủi ro.

Tiến trình phần mềm

- Mô hình tăng trưởng (Incremental model)



Tiến trình phần mềm

- Mô hình tăng trưởng (Incremental model)

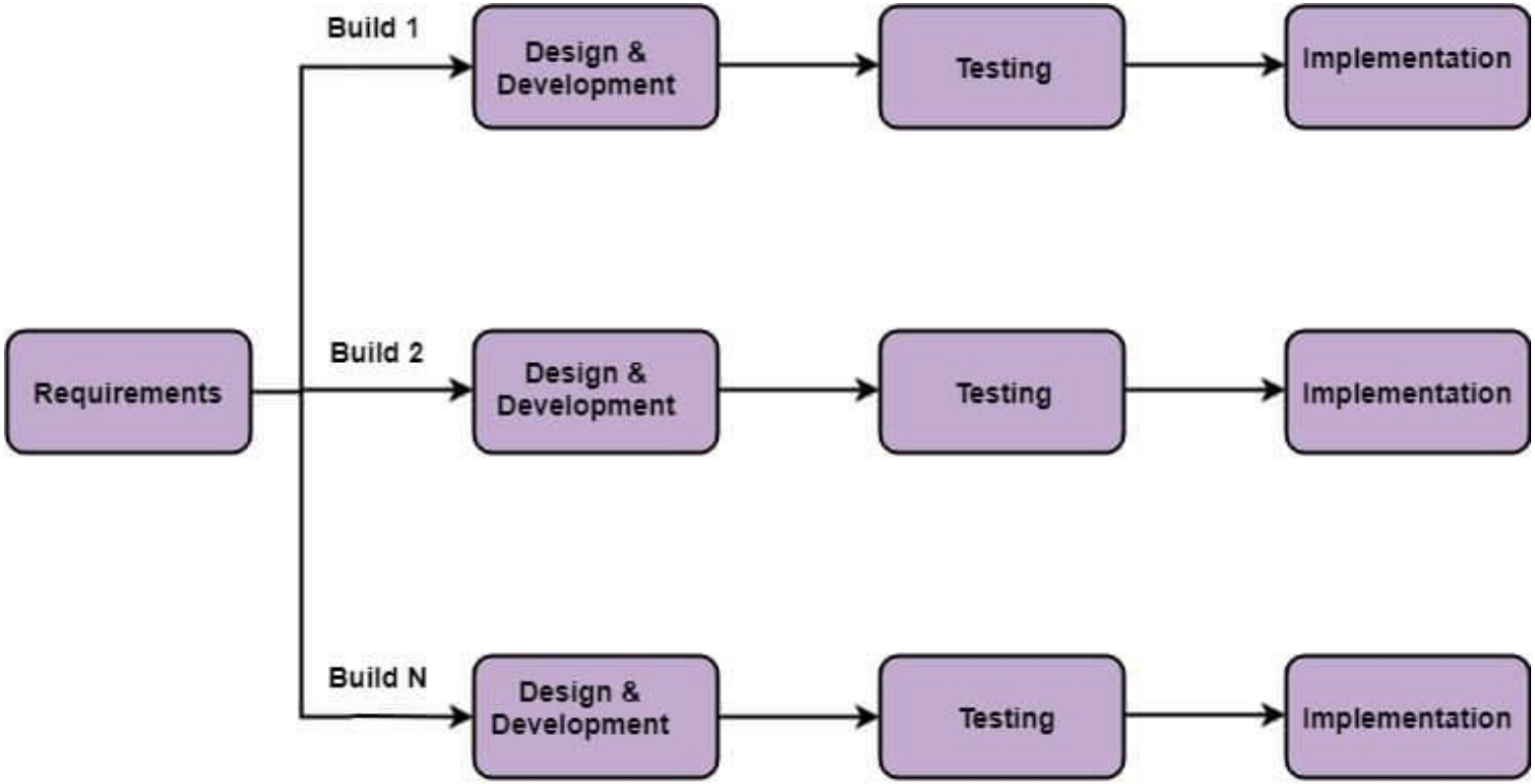


Fig: Incremental Model

Tiến trình phần mềm

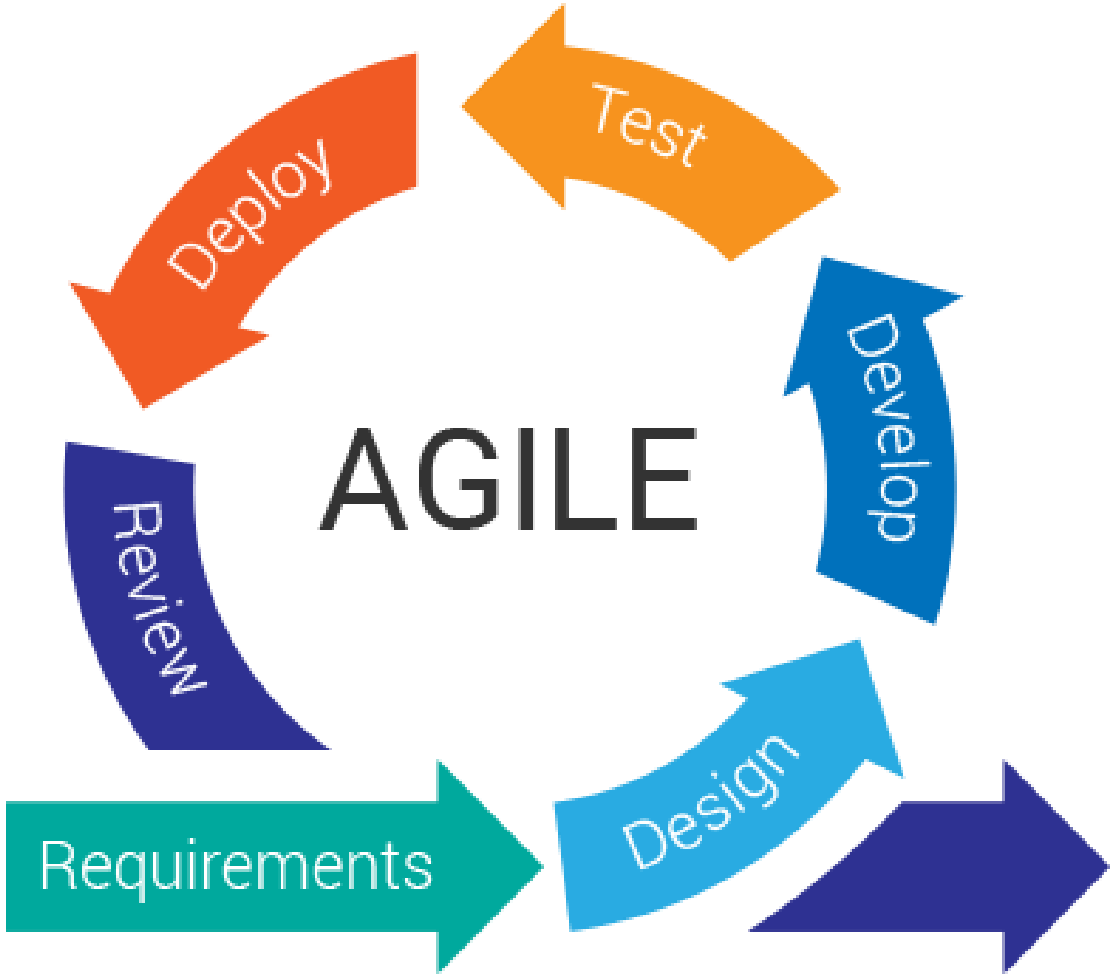
- Mô hình tăng trưởng (Incremental model)
  - Mô tả
    - Chu kỳ được chia thành các module nhỏ, dễ quản lý.
    - Mỗi module sẽ đi qua các yêu cầu về thiết kế, thực hiện… như 1 vòng đời phát triển thông thường.
  - Ứng dụng
    - Áp dụng cho những dự án có yêu cầu đã được mô tả, định nghĩa và hiểu một cách rõ ràng.
    - Khách hàng có nhu cầu về sản phẩm sớm.

Tiến trình phần mềm

- Mô hình tăng trưởng (Incremental model)
  - Ưu điểm
    - Phát triển nhanh chóng.
    - Mô hình này linh hoạt hơn, ít tốn kém hơn khi thay đổi phạm vi và yêu cầu.
    - Dễ dàng hơn trong việc kiểm tra và sửa lỗi.
  - Nhược điểm
    - Cần lập plan và thiết kế tốt.
    - Tổng chi phí là cao hơn so với mô hình thác nước.

Tiến trình phần mềm

- Mô hình Agile:



Tiến trình phần mềm

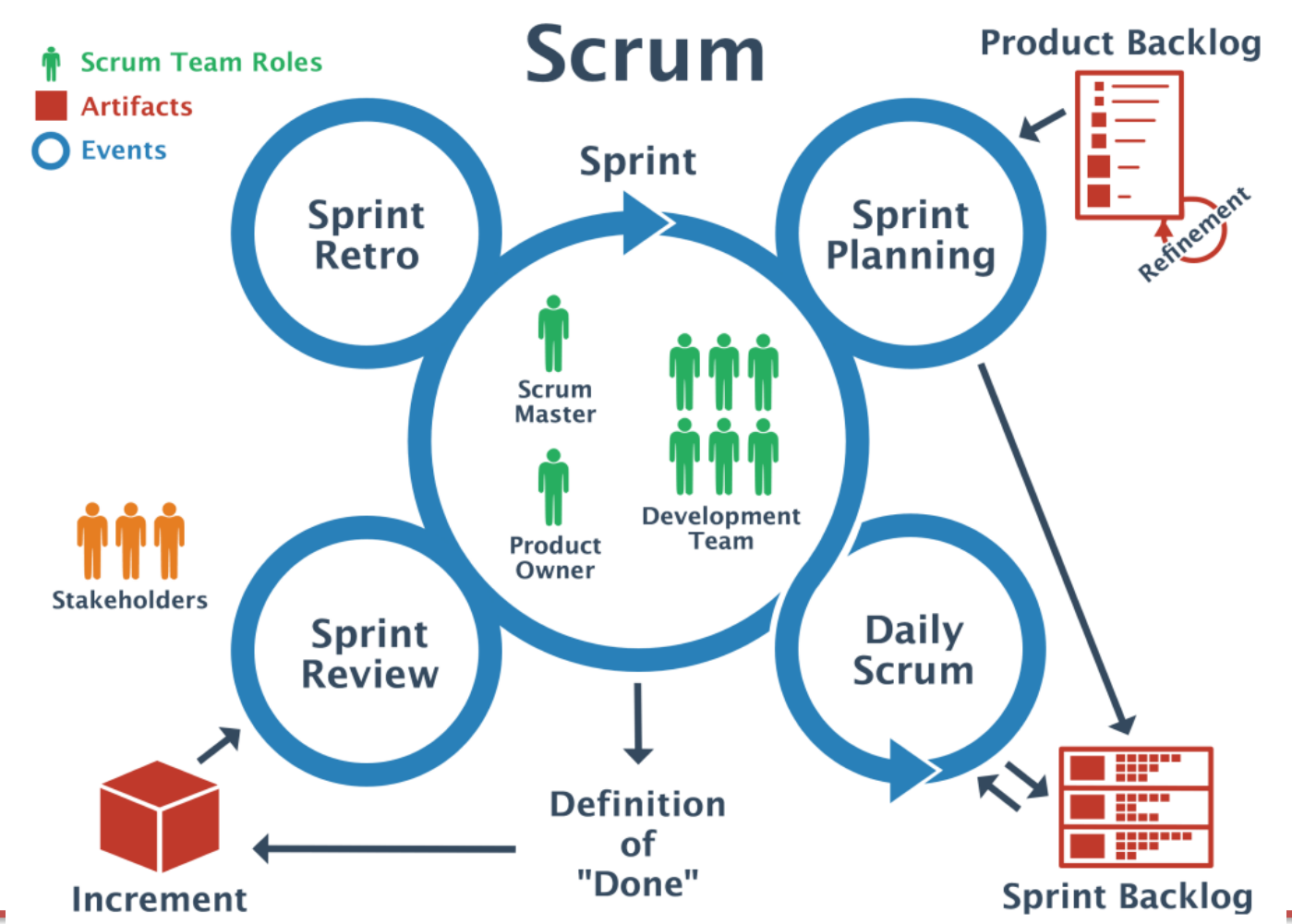
- Mô hình Agile:
  - Mô tả
    - Dựa trên mô hình iterative and incremental.
    - Các yêu cầu và giải pháp phát triển dựa trên sự kết hợp của các function.
    - Trong Agile, các tác vụ được chia thành các khung thời gian nhỏ để cung cấp các tính năng cụ thể cho bản phát hành cuối.
  - Ứng dụng
    - Có thể được sử dụng với bất kỳ loại hình dự án nào, nhưng cần sự tham gia và tính tương tác của khách hàng.
    - Sử dụng khi khách hàng yêu cầu chức năng sẵn sàng trong khoảng thời gian ngắn.

Tiến trình phần mềm

- Mô hình Agile:
  - Ưu điểm
    - Tăng cường tình thần làm việc nhóm và trao đổi công việc hiệu quả.
    - Các chức năng được xây dựng nhanh chóng và rõ ràng, dế quản lý.
    - Dễ dàng bổ sung, thay đổi yêu cầu.
    - Quy tắc tối thiểu, tài liệu dễ hiểu, dễ sử dụng.
  - Nhược điểm
    - Không thích hợp để xử lý các phụ thuộc phức tạp.
    - Có nhiều rủi ro về tính bền vững, khả năng bảo trì và khả năng mở rộng.
    - Cần một team có kinh nghiệm.
    - Phụ thuộc rất nhiều vào sự tương tác rõ ràng của khách hàng.
    - Chuyển giao công nghệ cho các thành viên mới trong nhóm có thể khá khó khăn do thiếu tài liệu.

Tiến trình phần mềm

- Mô hình Scrum:



Tiến trình phần mềm

- Mô hình Scrum:
  - Mô tả
    - Chia các yêu cầu ra làm theo từng giai đoạn. Mỗi 1 giai đoạn (sprint) chỉ làm 1 số lượng yêu cầu nhất định.
    - Mỗi một sprint thường kéo dài từ 1 tuần đến 4 tuần (ko dài hơn 1 tháng).
    - Đầu sprint sẽ lên kế hoạch làm những yêu cầu nào. Sau đó, sẽ thực hiện code và test. Cuối sprint là 1 sản phẩm hoàn thiện cả code lẫn test có thể demo và chạy được.
    - Hoàn thành sprint 1, tiếp tục làm sprint 2, sprint... cho đến khi hoàn thành hết các yêu cầu.
    - Trong mỗi 1 sprint thì sẽ có họp hàng ngày – daily meeting từ 15 – 20 phút. Mỗi thành viên sẽ báo cáo: Hôm qua tôi đã làm gì? Hôm nay tôi sẽ làm gì? Có gặp khó khăn gì không?
    - Scrum là mô hình hướng khách hàng (Customer oriented).

Tiến trình phần mềm

- Mô hình Scrum:
  - Tổ chức (Organization)
    - Tổ chức nhóm dự án và Roles: Vài trò.
    - Product Owner: Người sở hữu sản phẩm.
    - ScrumMaster: Người điều phối.
    - Development Team: Nhóm phát triển.
  - Tài liệu (Atifacts): đó chính là các kết quả đầu ra.
    - Product Backlog: Danh sách các chức năng cần phát triển của sản phẩm.
    - Sprint Backlog: Danh sách các chức năng cần phát triển cho mỗi giai đoạn.
    - Estimation: Kết quả ước lượng của team.
  - Qui trình (Process): Qui định cách thức vận hành của SCRUM.
    - Sprint Planning meeting: Hoạch định cho mỗi giai đoạn.
    - Review: Tổng kết cho mỗi giai đoạn.
    - Daily Scrum Meeting: Review hàng ngày.

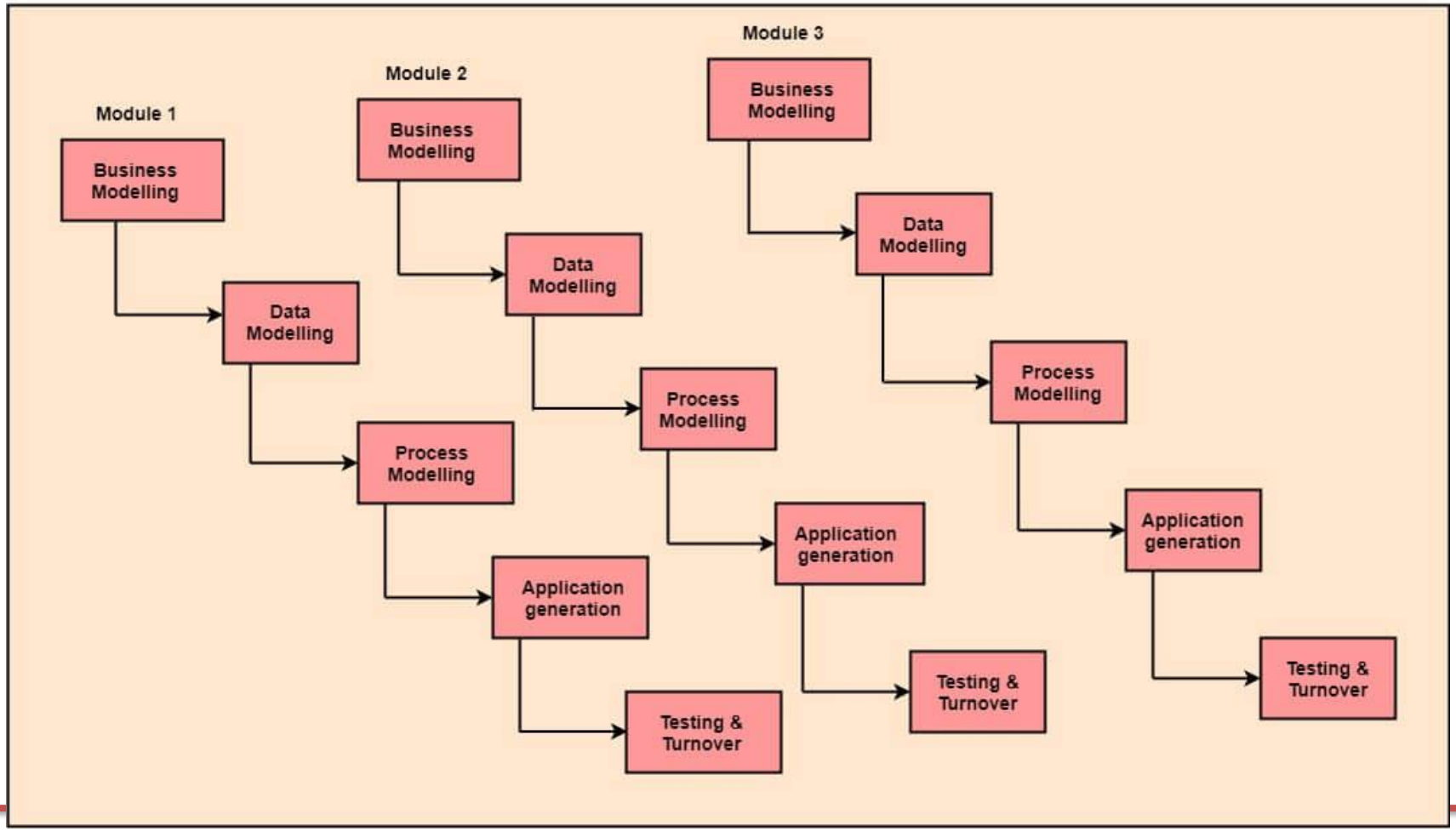
Tiến trình phần mềm

- Mô hình Scrum:
  - Ưu điểm
    - Một người có thể thực hiện nhiều việc ví dụ như dev có thể test.
    - Phát hiện lỗi sớm.
    - Có khả năng áp dụng được cho những dự án mà yêu cầu khách hàng không rõ ràng ngay từ đầu.
  - Nhược điểm
    - Trình độ của nhóm cần có một kỹ năng nhất định.
    - Phải có sự hiểu biết về mô hình aglie.
    - Khó khăn trong việc xác định ngân sách và thời gian.
    - Luôn nghe ý kiến phản hồi từ khách hàng và thay đổi theo nên thời gian sẽ kéo dài.
    - Vai trò của PO rất quan trọng, PO là người định hướng sản phẩm. Nếu PO làm không tốt sẽ ảnh hưởng đến kết quả chung.

Tiến trình phần mềm

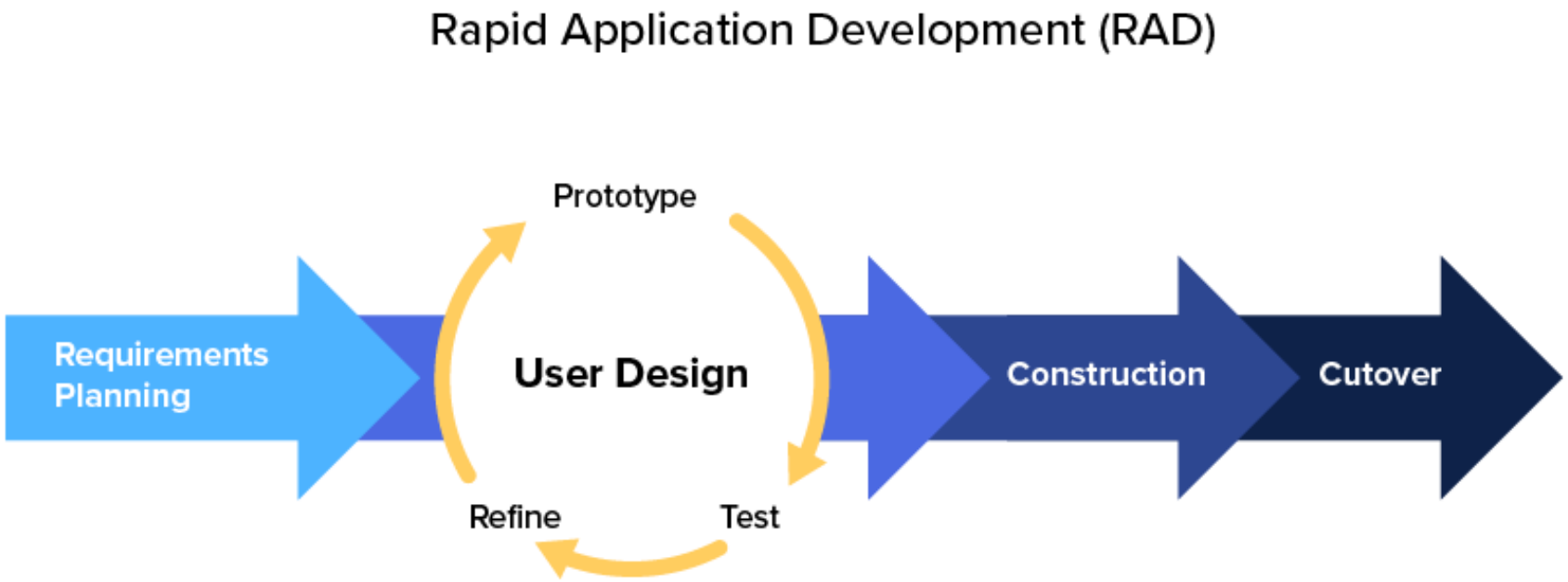
- Mô hình RAD:

Fig: RAD Model



Tiến trình phần mềm

- Mô hình RAD:



Tiến trình phần mềm

- Mô hình RAD:
  - Mô tả
    - Mô hình RAD là một phương pháp phát triển phần mềm sử dụng quy hoạch tối thiểu có lợi cho việc tạo mẫu nhanh.
    - Các module chức năng được phát triển song song như nguyên mẫu và được tích hợp để tạo ra sản phẩm hoàn chỉnh để phân phối sản phẩm nhanh hơn.
    - Đảm bảo rằng các nguyên mẫu được phát triển có thể tái sử dụng được.
  - Ứng dụng
    - Mô hình RAD có thể được áp dụng thành công cho các dự án:
    - Module hóa rõ ràng. Nếu dự án không thể được chia thành các module, RAD có thể không thành công.
    - RAD nên được sử dụng khi có nhu cầu để tạo ra một hệ thống có yêu cầu khách hàng thay đổi trong khoảng thời gian nhỏ 2-3 tháng.
    - Nên được sử dụng khi đã có sẵn designer cho model và chi phí cao.

Tiến trình phần mềm

- Mô hình RAD:
  - Ưu điểm
    - Giảm thời gian phát triển.
    - Tăng khả năng tái sử dụng của các thành phần.
    - Đưa ra đánh giá ban đầu nhanh chóng.
    - Khuyến khích khách hàng đưa ra phản hồi.
  - Nhược điểm
    - Trình độ của nhóm cần có một kỹ năng nhất định.
    - Chỉ những hệ thống có module mới sử dụng được mô hình này.

Tiến trình phần mềm

- Mô hình Big Bang
  - Sử dụng trong nhóm nhỏ, dự án nhỏ, không có quy trình rõ ràng

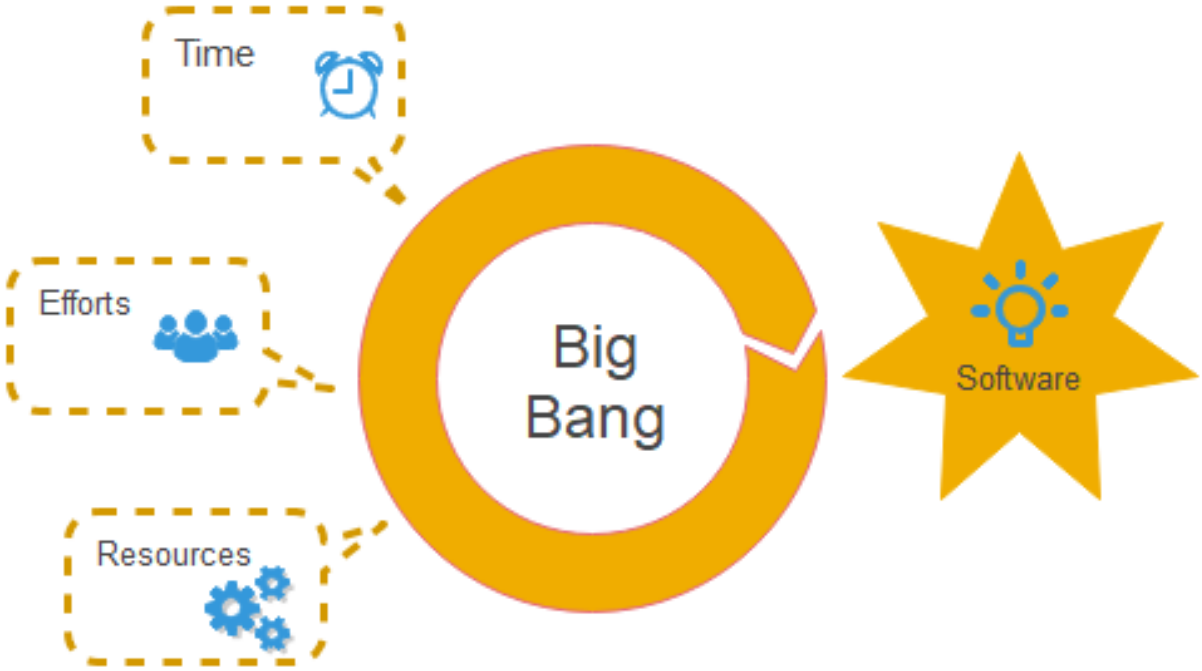


Fig. Big Bang Model

1.5. Một số khái niệm liên quan

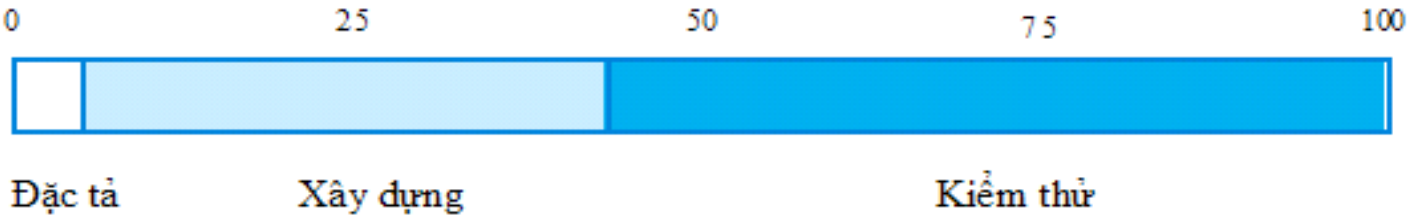
- Các chi phí trong công nghệ phần mềm
  - Để xây dựng một hệ thống phần mềm, thường phải đầu tư một khoản ngân sách khá lớn.
    - Theo thống kê cho thấy, chi phí cho việc xây dựng phần mềm chiếm một phần đáng kể của GNP ở tất cả các nước phát triển.
  - Chi phí phần mềm thường chiếm phần lớn chi phí của cả hệ thống máy tính.
    - Chi phí phần mềm trên máy PC thường lớn hơn chi phí phần cứng.
    - Chi phí phần mềm dành cho việc bảo trì phần mềm thường lớn hơn chi phí xây dựng phần mềm.
    - Đối với những hệ thống hoạt động trong thời gian dài, thì chi phí bảo trì thường lớn gấp nhiều lần so với chi phí xây dựng.

1.5. Một số khái niệm liên quan

- Các chi phí trong công nghệ phần mềm
  - Chi phí của công nghệ phần mềm gồm các khoản chi liên quan đến phát triển phần mềm
  - Chi phí phụ thuộc vào
    - Loại hệ thống được phát triển (đơn giản, phức tạp)
    - Yêu cầu đặt ra cho hệ thống (nhiều, ít)
    - Các thuộc tính về mức độ hoàn thiện của hệ thống (độ tin cây, an toàn, bảo mật)
    - Năng lực của tổ chức phát triển (nhân lực, công nghệ, công cụ, kỹ năng có được)
    - Loại tiến trình sử dụng

Các chi phí trong công nghệ phần mềm

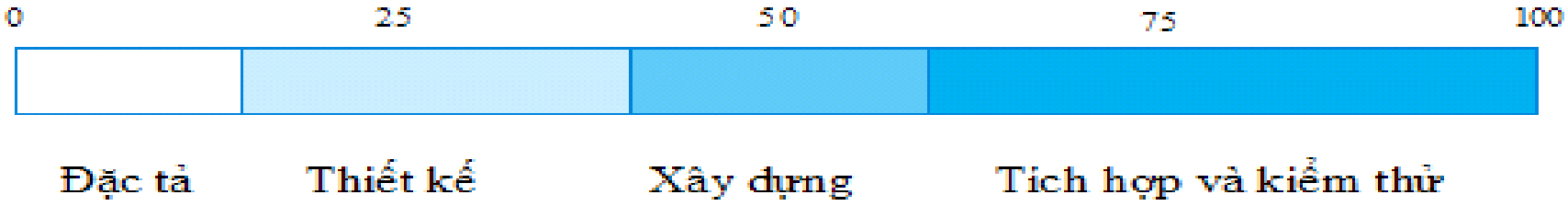
- Xấp xỉ 60% chi phí là chi phí xây dựng và 40% là chi phí kiểm thử.
- Đối với những phần mềm làm theo yêu cầu của khách hàng, chi phí mở rộng thường vượt quá chi phí xây dựng.
- Đối với những hệ thống có chu kỳ sống dài chi phí bảo trì có thể cao hơn chi phí phát triển nhiều lần
- Chi phí thay đổi tuỳ thuộc vào từng loại hệ thống được xây dựng và các yêu cầu về đặc điểm của hệ thống như:



Hình 1.1: Chi phí sản xuất phần mềm

Các chi phí trong công nghệ phần mềm

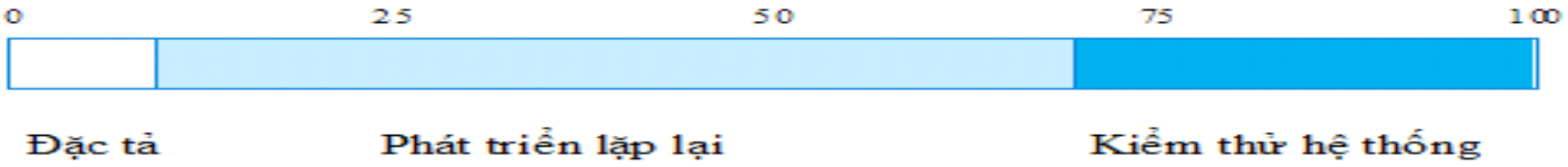
- Việc phân bổ chi phí cũng phụ thuộc vào mô hình phát triển hệ thống được sử dụng.
- Sau đây là bảng so sánh chi phí của 3 mô hình phổ biến nhất, thường được sử dụng:
  - Mô hình thác nước: Chi phí của các pha đặc tả, thiết kế, cài đặt, tích hợp và kiểm thử được xác định một cách riêng rẽ.



Hình 1.2: Phân bổ chi phí trong mô hình thác nước

Các chi phí trong công nghệ phần mềm

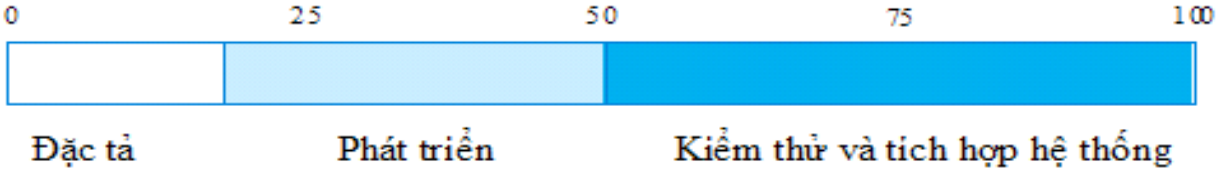
- Mô hình phát triển lặp lại:
  - Không thể phân biệt rõ chi phí cho từng pha trong quy trình.
  - Chi phí đặc tả giảm vì đây là đặc tả ở bậc cao.
  - Tại mỗi bước lặp, các pha trong quy trình xây dựng hệ thống được thực hiện lại nhằm thực hiện các yêu cầu hệ thống khác nhau ở từng bước lặp.
  - Sau khi đã thực hiện hết các bước lặp, phải có chi phí kiểm thử toàn bộ hệ thống.



Hình 1.3: Phân bổ chi phí trong mô hình phát triển lặp lại

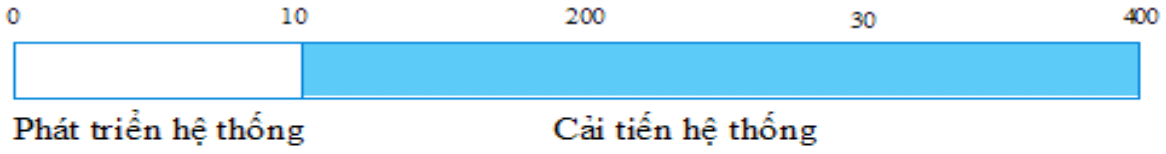
Các chi phí trong công nghệ phần mềm

- Mô hình công nghệ phần mềm hướng thành phần:
  - Chi phí phụ thuộc nhiều vào việc tích hợp và kiểm thử hệ thống.



Hình 1.4: Phân bổ chi phí trong công nghệ phần mềm hướng thành phần

- Ngoài chi phí xây dựng còn phải để một phần lớn chi phí phục vụ cho việc thay đổi phần mềm sau khi nó đã được đưa vào sử dụng. Chi phí cải tiến phần mềm thay đổi phụ thuộc vào từng loại phần mềm.



Hình 1.5: Phân bổ chi phí trên các hệ thống có chu kỳ sống dài

Công cụ và môi trường phát triển PM

- CASE (Computer Aided Software Engineering) tools.
- Các hệ thống CASE thường được sử dụng để hỗ trợ các hoạt động trong quy trình xây dựng phần mềm. Có hai loại CASE:
  - Upper-CASE: công cụ để hỗ trợ các hoạt động đầu tiên như đặc tả yêu cầu và thiết kế.
  - Lower-CASE: công cụ để hỗ trợ các hoạt động sau như lập trình, gỡ lỗi và kiểm thử.

Công cụ và môi trường phát triển PM

Giai đoạn phát triển	Công việc	Phần mềm
Phân tích	• Soạn thảo mô hình thế giới thực • Ánh xạ vào mô hình logic	
Thiết kế	• Soạn thảo mô hinh logic • Ánh xạ vào mô hình vật lý	
Cài đặt	• Quản lý các phiên bản • Lập trình • Biên dịch	
Kiểm thử	• Phát sinh tự động các bộ dữ liệu thử nghiệm • Phát hiện lỗi	
Xây dựng phương án	• Tạo lập phương án • Dự đoán rủi ro • Tính chi phí	
Lập kế hoạch	• Xác định các công việc • Phân công • Lập lịch biểu • Theo dõi thực hiện	

Nghề nghiệp

- Quy trình xây dựng phần mềm được thực hiện trong một môi trường chuyên nghiệp và đòi hỏi tuân thủ các nguyên tắc một cách chính xác.
  - Những kỹ sư phần mềm phải coi công việc của mình là trách nhiệm to lớn, chứ không đơn thuần chỉ là việc ứng dụng kỹ thuật.
  - Kỹ sư phần mềm phải ứng xử trung thực và cách làm của họ phải rất chuyên nghiệp và đúng quy tắc.
- Vấn đề về tính chuyên nghiệp và đúng quy tắc đối với kỹ sư phần mềm quan trọng tới mức một số tổ chức ở Mỹ đã hợp tác để phát triển bản Code of Ethics gồm 8 quy tắc liên quan đến ứng xử và cách ra quyết định của các kỹ sư phần mềm chuyên nghiệp.

Nghề nghiệp

- Sự tin cậy: kỹ sư phần mềm phải tạo được sự tin cậy từ phía nhân viên và khách hàng.
- Năng lực: kỹ sư phần mềm không nên trình bày sai khả năng của mình, không nên nhận những công việc vượt quá khả năng của mình.
- Các quyền về tài sản trí tuệ: kỹ sư phần mềm nên quan tâm về các tài sản trí tuệ được bảo hộ như: bằng sáng chế, quyền tác giả… để đảm bảo rằng tất cả tài sản trí tuệ của nhân viên và khách hàng đều được bảo hộ.
- Lạm dụng máy tính: kỹ sư phần mềm không nên sử dụng các kỹ năng của mình để gây ảnh hưởng tới người khác.
  - Lạm dụng máy tính có thể được hiểu là những việc tầm thường (Ví dụ: chơi điện tử trên máy tính của người khác) đến những vấn đề nghiêm trọng (Ví dụ: phát tán virus).

Nghề nghiệp

- Kỹ năng của kỹ sư phần mềm
  - Phân tích thiết kế hệ thống, mô hình hóa
  - Sử dụng bản mẫu (prototype)
  - Chọn phần cứng, phần mềm
  - Quản lý cấu hình, lập sơ đồ và kiểm soát tiến trình
  - Lựa chọn ngôn ngữ và phát triển chương trình
  - Xác định, đánh giá, cài đặt, chọn phương pháp, công cụ CASE
  - Kiểm tra, kiểm thử (testing)
  - Lựa chọn và sử dụng kỹ thuật bảo trì phần mềm
  - Đánh giá và quyết định khi nào loại bỏ và nâng cấp các ứng dụng

🕮 Nghề nghiệp

- Tester (người kiểm thử):
  - Nghiên cứu yêu cầu của khách hàng, các chi tiết yêu cầu về phần mềm và cách đánh giá.
  - Lập kế hoạch kiểm thử (test plan) dựa trên phân tích rủi ro, mức độ quan trọng, tần suất sử dụng và xác định các yếu tố liên quan như: phương pháp, đo lường, nguồn lực, điều kiện kiểm tra…
  - Tạo test cases/specs/scripts.
  - Tiến hành test dựa trên test case và ghi báo cáo (thành công hoặc lỗi).
  - Log các lỗi tìm được và lập báo cáo (test report).
  - Phân tích các biến đổi (change), cập nhật các tài liệu kiểm thử (Test plan, test cases…).

🕮 Nghề nghiệp

- Tester (người kiểm thử): đối với kiểm thử phần mềm có 2 năm kinh nghiệm trở lên (Senior Tester):
  - Lập kế hoạch kiểm thử.
  - Phụ trách một nhóm tester, xem xét các test case do các tester tạo ra, tính toán và phân tích các chỉ số liên quan đến việc kiểm thử.
  - Nghiên cứu automation test tools và áp dụng vào test dự án
  - Đề xuất cải tiến qui trình kiểm thử, lập lưu đồ, hướng dẫn, biểu mẫu… để thực hiện và quản lý việc kiểm thử.

🕮 Nghề nghiệp

- QA (Quality Assurance - bảo đảm chất lượng):
  - Các nhân viên QA sẽ kiểm tra chất lượng công việc trong cả quá trình sản xuất, ví dụ xem xét tài liệu yêu cầu, tài liệu thiết kế hay xem xét mã nguồn. Họ cũng xem xét các ý kiến phản hồi của khách hàng và cách thức các đội dự án xử lý vấn đề.
  - QA thu thập thông tin của các dự án, các chỉ số báo chất lượng của dự án, từ đó vẽ nên bức tranh chung về tình hình chất lượng sản xuất của cả công ty
  - Công việc của QA vừa mang tính chất giám sát, vừa mang tính hỗ trợ cho hoạt động của dự án.

🕮 Nghề nghiệp

- Người Quản lý dự án (Project Manager)
  - Thỏa mãn các mục tiêu phạm vi, thời gian, chi phí và chất lượng dự án
  - Thỏa mãn được nhu cầu và mong đợi của mọi người có liên quan hoặc bị ảnh hưởng bởi dự án.
  - PM thực hiện các nhiệm vụ chính sau: lập kế hoạch, tạo lịch biểu, cộng tác, và giao tiếp để đạt mục tiêu của dự án
  - 97% sự thành công của dự án là nhờ vào PM

Nghề nghiệp

- Người Quản lý dự án (Project Manager): Các kỹ năng cần thiết của người Quản lý dự án
  - Kỹ năng giao tiếp: lắng nghe, thuyết phục.
  - Kỹ năng tổ chức: lập kế hoạch, xác định mục tiêu, phân tích.
  - Kỹ năng xây dựng nhóm: thấu hiểu, thúc đẩy, tinh thần đồng đội.
  - Kỹ năng lãnh đạo: năng động, có tầm nhìn, biết giao nhiệm vụ, lạc quan.
  - Kỹ năng đối phó: linh hoạt, sáng tạo, kiên trì, chịu đựng.
  - Kỹ năng công nghệ: kinh nghiệm, kiến thức về dự án.

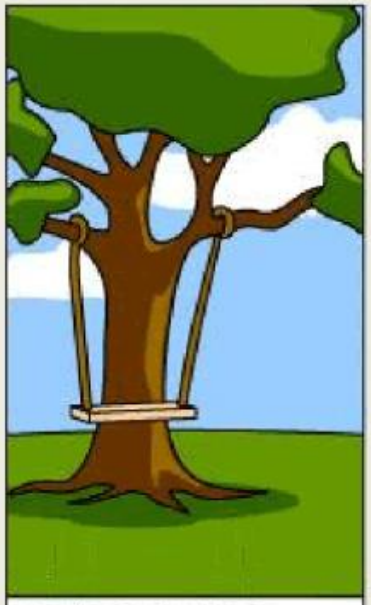
Các nhận thức sai lầm

- Tôi dễ dàng biến đổi phần mềm.
  - Thực tế: những thay đổi yêu cầu là nguyên nhân chính làm giảm giá trị phần mềm.
- Tôi có thể giải quyết vấn đề lịch biểu bằng cách thêm người.
  - Thực tế: nó có thể đòi hỏi gia tăng nỗ lực phối hợp làm giảm hiệu suất.
  - Định luật Brooks: “Thêm người vào một Dự án phần mềm bị chậm thì chỉ làm cho nó chậm hơn”.
- Chỉ cần xem xét sơ lược các yêu cầu, chúng tôi có thể bắt đầu viết code.
  - Thực tế: đó thường là nguyên nhân chính gây ra lỗi.

Các nhận thức sai lầm

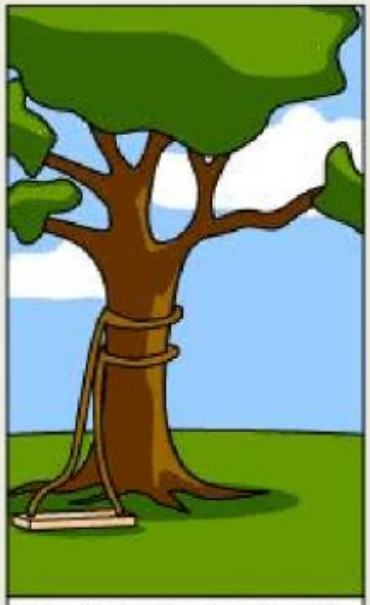
- Phần quan tâm hầu như duy nhất là mã
  - Tài liệu, thông tin kiểm thử và cấu hình phần mềm cũng là những phần then chốt trong việc chuyển giao.
- Lập trình giỏi thì hầu như là sẽ tạo được phần mềm chất lượng.
  - Những dự án phần mềm được hoàn tất bởi những nhóm, không phải cá nhân, để thành công cần nhiều hơn là tạo mã.
- Tạo mã là công việc nặng nhọc nhất.
  - Tạo mã: ít hơn 10-30% công sức.

How the customer explained it



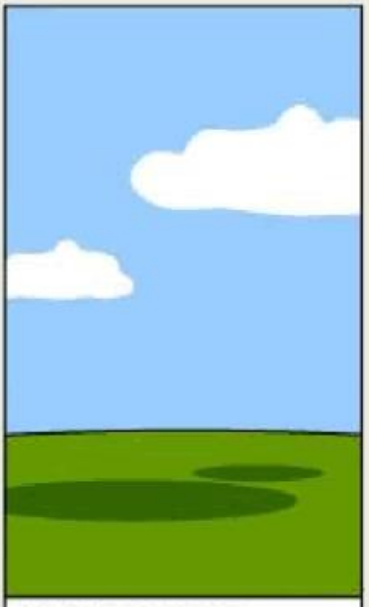
How the Project Leader understood it

How the Analyst designed it

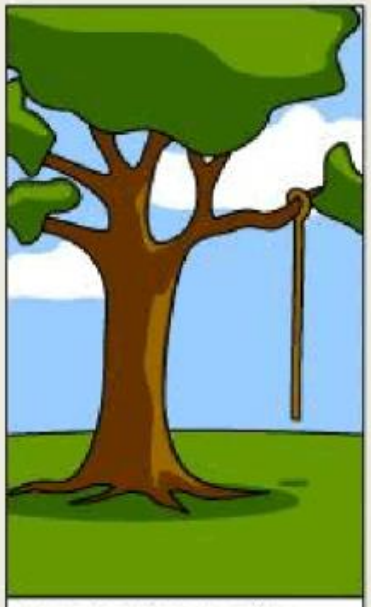


How the Programmer wrote it

How the Business Consultant described it

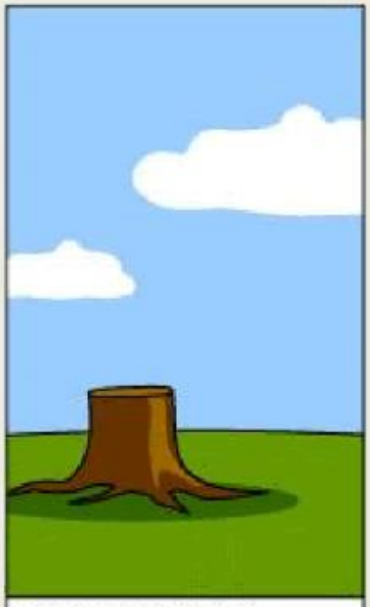


How the project was documented



What operations installed

How the customer was billed



How it was supported

What the customer really needed

🕮 Câu hỏi thảo luận