NHẬP MÔN
CÔNG NGHỆ PHẦN MỀM

TÀI LIỆU

DÀNH CHO SINH VIÊN CÔNG NGHỆ THÔNG TIN

TRẦN ĐÌNH QUẾ

GIỚI THIỆU

Mục tiêu của môn Công nghệ phần mềm là cung cấp cho sinh viên những kiến thức cơ bản về tất cả mọi hoạt động liên quan đến phát triển phần mềm và kiến thức cơ bản về UML trong phát triển phần mềm. Qua môn học này sinh viên có kỹ năng sử dụng công cụ phần mềm để thực hiện các pha trong quá trình phát triển phần mềm và qua đó nâng cao năng lực làm việc nhóm và kỹ năng mềm. Sinh viên tham dự lớp và thực hành đầy đủ đặc biệt tích cực tham gia thảo luận trình bày trên lớp là yêu cầu quan trọng.

Nội dung bao gồm các kiểu hệ thống thông tin, các mô hình phát triển phần mềm, lập kế hoạch và quản lý dự án; các pha phát triển phần mềm từ xác định yêu cầu, phân tích, thiết kế đến lập trình – tích hợp; các kiến thức cơ bản về mô hình phần mềm với UML.

Mở đầu: Các đặc trưng của phần mềm**;** Các dạng phần mềm**;** Các hoạt động trong phát triển phần mềm**;** Tiến hóa trong phát triển phần mềm

Chương 2: Các pha trong phát triển phần mềm.

Các tác nhân trong quá trình phát triển phần mềm**;** Pha xác định yêu cầu**;** Pha phân tích**;** Pha thiết kế**;** Pha cài đặt và tích hợp**;** Pha bảo trì

Chương 3: Các mô hình vòng đời phần mềm

Mô hình xây - sửa; Mô hình thác nước; Mô hình bản mẫu nhanh; Mô hình tiến hoá; Mô hình RUP; Mô hình xoắn ốc; So sánh các mô hình

Chương 4: Kiểm chứng

Vấn đề chất lượng phần mềm; Kiểm chứng phần mềm; Các phương pháp kiểm chứng; Công cụ kiểm chứng

Chương 5: Lập kế hoạch và ước lượng

Vấn đề lập kế hoạch và ước lượng dự án phần mềm; Ước lượng thời gian và chi phí; Các thành phần của việc lập kế hoạch dự án phần mềm; Lập kế hoạch cho các dự án phần mềm hướng đối tượng.

Chương 6: Xác định yêu cầu

Các kỹ thuật xác định yêu cầu; Bản mẫu nhanh; Đặc tả dựa trên bản mẫu nhanh; Sử dụng lại bản mẫu; Đặc tả với bản mẫu; Kiểm thử pha yêu cầu

Chương 7: Các phương pháp phân tích truyền thống

Viết tài liệu pha đặc tả; Đặc tả phi hình thức; Các kỹ thuật đặc tả nửa hình thức; Mô hình quan hệ thực thể; Máy trạng thái hữu hạn; Các kỹ thuật đặc tả hình thức; So sánh các kỹ thuật đặc tả; Kiểm thử pha đặc tả

Chương 8: Phân tích hướng đối tượng

Tổng quan về phân tích hướng đối tượng; Mô hình use case; Mô hình lớp; Mô hình hành động; Kiểm thử pha phân tích hướng đối tượng

Chương 9: Thiết kế

Tổng quan về pha thiết kế; Thiết kế hướng hành động; Phân tích dòng dữ liệu; Thiết kế hướng đối tượng

Chương 10: Cài đặt và tích hợp

Các phương pháp cài đặt và tích hợp; Kiểm thử pha cài đặt và tích hợp; Kiểm thử sản phẩm; Kiểm thử chấp nhận

Chương 11: Bảo trì

Pha bảo trì**;** Bảo trì hệ phần mềm hướng đối tượng

TÀI LIỆU THAM KHẢO

[1] Object-Oriented and Classical Software Engineering, Stephen R. Schach, Seventh Edition, Mc Graw Hill, 2008.

[2] Giáo trình nhập môn UML, Huỳnh Văn Đức, Hoàn Thiện Ngân, NXB Lao động Xã hội, 2003.

MỤC LỤC

CHƯƠNG 1: MỞ ĐẦU

1.1 ĐẶC TRƯNG CỦA PHẦN MỀM

- Phần mềm không mòn
- Phần mềm được phát triển mà không được sản xuất theo nghĩa thông thường
- Mặc dù công nghiệp phần mềm đang hướng đến phát triển dựa trên thành phần nhưng phần lớn phần mềm phát triển dựa theo yêu cầu của khách hàng.
- Cho đến nay những đặc trưng của phần mềm vẫn còn là vấn đề tranh cãi. Chính điều này thể hiện sự chưa trưởng thành của ngành học CÔNG NGHỆ PHẦN MỀM.

1.2 CÁC KIỂU PHẦN MỀM

- **Phần mềm hệ thống:** Tập hợp các Chương trình được viết để phục vụ các chương trình khác, tương tác với phần cứng (ví dụ, biên dịch, trình soạn thảo, HĐH…)
- **Phần mềm thời gian thực:** Phần mềm điều phối/phân tích kiểm soát, đáp ứng thời gian
- **Phần mềm nghiệp vụ:** Các phần mềm tính lương, kế toán, quản lý kho…
- **Phần mềm khoa học và công nghệ.** Các ứng dụng trong thiên văn, sinh học phân tử, điều khiển tàu con thoi,…
- **Phần mềm nhúng:** Nằm trong bộ nhớ chỉ đọc, điều khiển các sản phẩm và hệ thống
- **Phần mềm máy tính cá nhân:** Xuất văn bản, đồ họa máy tính…
- **Phần mềm trên Web**
- **Phần mềm trí tuệ nhân tạo:** Dựa trên những kỹ thuật của Trí tuệ nhân tạo như hệ chuyên gia.

Nhận xét: Hiện nay web được xem là môi trường phổ biến để xây dựng giao diện với người sử dụng cho nhiều hệ thống phần mềm trên mạng.

1.3 KHÍA CẠNH LỊCH SỬ

- Năm 1967 nhóm NATO đưa ra thuật ngữ Công nghệ phần mềm (Software Engineering).
- Năm 1968 Hội nghị Software Engineering ở Garmisch, Đức đưa ra mục đích là giải quyết "Cuộc khủng hoảng phần mềm":
  - Phần mềm hoàn thành không đúng thời hạn
  - Chi phí vượt dự toán ban đầu
  - Phần mềm còn nhiều lỗi
- Tại sao không thể sử dụng kỹ nghệ xây cất như xây dựng cầu để xây dựng các hệ điều hành?
  - Thái độ đối với việc sập cầu/hệ điều hành
  - Thông tin về CNPM thường không đầy đủ, không chắc chắn
  - Độ phức tạp

- Bảo trì

Công nghệ phần mềm không thể xem giống như các kỹ nghệ thông thường khác.

1.4 KHÍA CẠNH KINH TẾ

- CNPM và khoa học máy tính (tương tự như hóa học và kỹ nghệ hóa)
  - Khoa học có phần thực nghiệm và lý thuyết: Phần thực nghiệm của Hóa học là thí nghiệm còn của khoa học máy tính là lập trình.
  - Khoa học máy tính nghiên cứu những sách khác nhau để tạo ra phần mềm. Nhưng kỹ sư phần mềm chỉ quan tâm kỹ thuật có ý nghĩa kinh tế.
  - Ví dụ: Phương pháp mã hóa mới KTmới (lập trình hướng thành phần) nhanh hơn phương pháp đang sử dụng hiện thời KTcũ (lập trình hướng đối tượng) là 10%. Chúng ta nên sử dụng phương pháp mới hay không?
    Câu trả lời thông thường là: Tất nhiên! Câu trả lời Công nghệ phần mềm: xét hiệu quả của KTmới.

1.5 KHÍA CẠNH BẢO TRÌ

- Vòng đời phần mềm: Một loạt các pha mà phần mềm phải trải qua từ khám phá các khái niệm đến loại bỏ hoàn toàn.
- Mô hình vòng đời
  - Pha yêu cầu
  - Pha đặc tả
  - Pha thiết kế
  - Pha cài đặt
  - Pha tích hợp
  - Pha bảo trì
  - Loại bỏ
- **Bảo trì:** Mọi thay đổi đối với sản phẩm một khách hàng đã đồng ý sản phẩm thỏa mãn tài liệu đặc tả.
- Phần mềm tốt được bảo trì khác với phần mềm tồi bị loại bỏ
- Các dạng bảo trì"
  - Bảo trì sửa lỗi [17,5%]: sửa chữa lỗi nhưng đặc tả không đổi
  - Bảo trì nâng cao: sửa chữa theo thay đổi của đặc tả. Bảo trì hoàn thiện [60,5%]: thêm chức năng để cải tiến sản phẩm. Bảo trì thích nghi [18%]: thay đổi phần mềm để đáp ứng thay đổi của môi trường như quy định chính phủ, CPU, công nghệ mới…

Ví dụ 1: Tỷ lệ thuế GTGT thay đổi từ 6% đến 7%

- C++

const float *salesTax =6.0;*

- JAVA

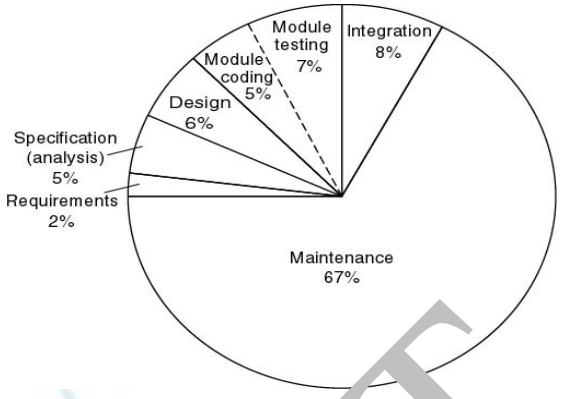
public static float *salesTax = 6.0;*

Ví dụ 2 Tổ chức y tế thay đổi ⇒ Hoạt động thay đổi

Ví dụ 3 Các hệ thời gian thực/hệ nhúng

Thế giới thực thay đổi ⇒ hệ thay đổi

- Chi phí cho các pha: Nguồn dữ liệu của các năm 1976-1981



- Hewlett-Packard (1992)
  - 60-80% nguồn nhân lực để phát triển và nghiên cứu dành cho bảo trì
  - 40-60% chi phí phần mềm dành cho bảo trì
- Nhiều tổ chức dành 80% thời gian và công sức chop ha bảo trì [Yourdon, 1996]

Kết luận: Bảo trì là pha tốn kém nhiều thời gian và chi phí

1.6 KHÍA CẠNH PHÂN TÍCH VÀ THIẾT KẾ

- 60 đến 70% lỗi của phần mềm là những lỗi do đặc tả và thiết kế
- Dữ liệu của Kelly, Sherif and Hops [1992] về chương trình không gian liên hành tinh của Nasa
  - 1,9 lỗi trên một trang đặc tả
  - 0,9 lỗi trên một trang thiết kế
  - 0,3 lỗi trên một trang chương trình nguồn
- Dữ liệu của Bhandari et al. [1994]: Lỗi trong cuối pha thiết kế của phiên bản mới của sản phẩm
  - 13% lỗi từ phiên bản trước
  - 16% lỗi trong pha đặc tả mới
  - 71% lỗi trong pha thiết kế mới

Kết luận: Xây dựng những kỹ thuật sinh ra thiết kế và đặc tả tốt hơn là một vấn đề quan trọng trong công nghệ phần mềm.

Chi phí cho việc phát hiện và sửa chữa lỗi:

1.7 KHÍA CẠNH LẬP TRÌNH NHÓM

Phần cứng rẻ, khả năng tăng, kích thước giảm

Phần mềm lớn, đắt. Nhiều phần mềm quá lớn nên một người không thể phát triển được trong thời gian có hạn.

Ví dụ: Lan và Minh viết code cho hai Mô đun p và q với mô đun p gọi q. Khi viết p Lan đã viết một hàm gọi q với 5 đối số. Minh cũng code q với 5 đối số nhưng thứ tự khác Lan.

- Compiler của C không thể phát hiện sai sót đó
- Java chỉ có thể phát hiện khi biến khác kiểu nhưng nếu cùng kiểu thì không thể phát hiện

1.8 PHƯƠNG PHÁP HƯỚNG ĐỐI TƯỢNG

Trước năm 1975, phần lớn các tổ chức phần mềm có những kỹ thuật riêng, cá nhân có cách làm việc riêng. Giữa những năm 1975-1985: Phương pháp cấu trúc đạt nhiều thành công ban đầu. Kỹ thuật cấu trúc phù hợp với chương trình 5.000 đến 50.000 dòng mã.

Hạn chế của phương pháp cấu trúc:

- Không phù hợp với phần mềm lớn (> 50.000 dòng lệnh)
- 80% chi phí và sức lực dành cho bảo trì nhưng cách tiếp cận của phương pháp cấu trúc không giải quyết được vấn đề này.

Đặc trưng của phương pháp cấu trúc:

- Hướng hành động (máy trạng thái hữu hạn, sơ đồ dòng dữ liệu); hay
- Hướng dữ liệu (sơ đồ quan hệ thực thể, phương pháp Jackson);
- Không cả hai

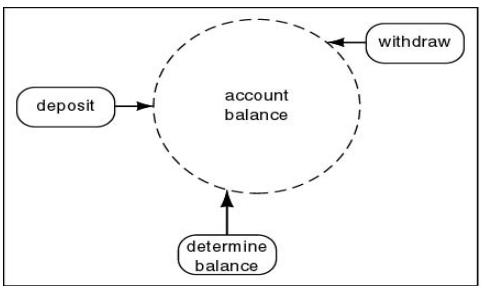
Phương pháp hướng đối tượng: Cả hai dữ liệu và hành động đều quan trọng như nhau

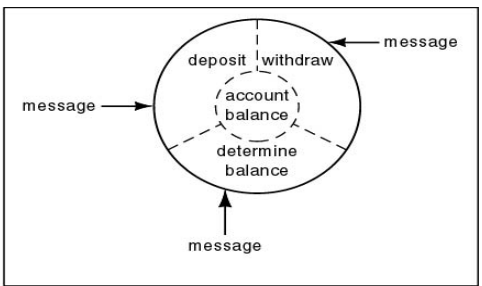
Đối tượng: Thành phần phần mềm kết hợp cả hai dữ liệu và các hành động thực hiện trên dữ liệu đó.

Ví dụ: Quản lý trương mục ngân hàng

- Dữ liệu: tiền gửi vào, rút ra
- Hành động: gửi vào, rút ra, xác định tiền gửi vào rút ra

Phương pháp lập trình hướng đối tượng khác phương pháp cấu trúc





- Ẩn dấu thông tin
- Thiết kế dựa trên trách nhiệm
- Ảnh hưởng đối với bảo trì và phát triển

- **Phương pháp hướng đối tượng:** Một đối tượng gửi message đến một đối tượng khác để yêu cầu hành động/phương pháp.
- **Từ phân tích đến thiết kế**

Structured Paradigm	**Object-OrientedParadigm**
1. Requirements phase	1. Requirements phase
2. Specification (analysis) phase	2′. Object-oriented analysis phase
3. Design phase	3′. Object-oriented design phase
4. Implementation phase	4′. Object-oriented programming phase
5. Integration phase	5. Integration phase
6. Maintenance phase	6. Maintenance phase
7. Retirement	7. Retirement

- Phương pháp cấu trúc: Tạo "khoảng cách" giữa phân tích (what) và thiết kế "how"
- Phương pháp hướng đối tượng: (chuyển pha "êm dịu" nên giảm được sai sót) Các đối tượng được đưa vào ngay từ đầu của vòng đời. Đối tượng được đề xuất trong pha thiết kế, được xây dựng trong pha thiết kế, được mã hóa trong pha lập trình.
- Phân tích hệ thống: Xác định điều cần phải làm WHAT
- Thiết kế xác định cách làm HOW: thiết kế kiến trúc xác định các mô đun và thiết kế chi tiết từng mô đun.

- **Lợi ích của HĐT:**
  - Phân tích hướng đối tượng: Xác định cái gì phải thực hiện và xác định các đối tượng
  - Thiết kế hướng đối tượng: Xác định cách thực hiện hành động và xây dựng các đối tượng.
- **So sánh phương pháp cấu trúc và phương pháp hướng đối tượng**

Structured Paradigm	Object-Oriented Paradigm
2. Specification (analysis) phase • Determine what the product is to do	2′. Object-oriented analysis phase • Determine what the product is to do • Extract the objects
3. Design phase • Architectural design (extract the modules) • Detailed design	3′. Object-oriented design phase • Detailed design
4. Implementation phase • Implement in appropriate programming language	4′. Object-oriented programming phase • Implement in appropriate object-oriented programminglanguage

Kết luận:

- Công nghệ phần mềm: ngành học nhằm nghiên cứu cách sản xuất ra phần mềm thỏa mãn nhu cầu khách hàng, không có lỗi, giao đúng thời hạn và trong chi phí hạn định.
- Công nghệ phần mềm xem xét các pha của vòng đời phần mềm và phân phối các khía cạnh khác nhau của tri thức con người từ kinh tế, kỹ thuật đến khoa học xã hội, tâm lý…
- Phương pháp hướng đối tượng được xem là cách tiếp cận phổ biến trong phát triểnphần mềm hiện nay.

CHƯƠNG 2: CÁC PHA PHÁT TRIỂN PHẦN MỀM

2.1 TIẾN TRÌNH THÀNH PHẦN

- Tiến trình phần mềm là "phương cách" sản xuất ra phần mềm. Tiến trình phần mềm nghiên cứu các cách kết hợp:
  - Mô hình vòng đời
  - Các công cụ CASE
  - Các cá nhân xây dựng phần mềm
  - Các công nghệ

  Tiến trình phần mềm = Khía cạnh kỹ thuật + Khía cạnh quản lý
- Các tổ chức khác nhau có những tiến trình phần mềm khác nhau.
  - Khâu viết tài liệu (quan trọng/không quan trọng)
  - Chi phí dành cho kiểm thử (1/2 chi phí/không quan trọng)
  - Tập trung khaua nghiên cứu, phát triển phần mềm. Khâu bảo trì dành cho người khác.
- Vì sao như vậy?
  - Thiếu kỹ năng về kỹ nghệ phần mềm
  - Nhiều người quản lý phần mềm thiếu kiến thức về bảo trì và phát triển phần mềm (do đó trễ hạn!)
  - Quan điểm về quản lý (giao đúng hạn hay kiểm tra kỹ trước khi giao)
  - Phụ thuộc vào cá nhân

Có pha kiểm thử không?

- KHÔNG có pha kiểm thử. Vì sao? Kiểm thử là hoạt động thực hiện trong mọi pha của sản xuất phần mềm.
- Check = test
- Testing: Thương được hiểu sau khi coding
- Kiểm tra (Varification): Thực hiện vào cuối mỗi pha
- Kiểm chứng (Validation): Thực hiện trước khi giao sản phẩm cho khách hàng

Có pha viết tài liệu không?

- KHÔNG có pha viết tài liệu
- Mọi pha phải được viết tài liệu trước khi khởi đầu một pha mới.
- Một số lý do:
  - Tài liệu bị hoãn lại thì sẽ không bao giờ hoàn thành
  - Cá nhân chịu trách nhiệm trong pha trước có thể đã chuyển sang bộ phận khác.

- Sản phẩm thường xuyên thay đổi trong khi phát triển vì thế ta cần tài liệu để ghi lại điều này. Ví dụ, thiết kế thường phải sửa đổi trong khi cài đặt. Việc sửa đổi như vậy chỉ có thể thực hiện được khi có tài liệu của nhóm thiết kế.

Kiểm thử và viết tài liệu được tiến hành trong mọi pha của tiến trình phần mềm.

2.2 SQA LÀ GÌ?

- Nhóm đảm bảo chất lượng phần mềm (SQA: Software quality assurance):
  - Có trách nhiệm đảm bảo sản phẩm được xây dựng đúng (theo đặc tả) và theo đơn đặt hàng.
  - Nhóm SQA phải đóng đúng vai trò ngay từ đầu của tiến trình và hoạt động trong mọi pha của tiến trình phần mềm.
  - SQA kiểm tra với khách hàng xem phiên bản cuối cùng thỏa mãn hoàn toàn chưa.

2.3 PHA YÊU CẦU

- Tiến trình phát triển phần mềm bắt đầu khi khách hàng tiếp xúc với công ty phần mềm và cho rằng: Phần mềm có khả năng thích hợp với khách hàng và giá thành hợp lý.
- Khám phá khái niệm:
  - Trong lần gặp đầu tiên, khách hàng phát hợ sản phẩm mà họ hình dung. Theo quan điểm của người phát triển, mô tả này không rõ ràng, không hợp lý, mâu thuẫn hay không thể xây dựng phần mềm như thế được.
  - Người phát triển xác định nhu cầu và ràng buộc của khách hàng

Kiểm thử pha yêu cầu:

- Làm bản mẫu nhanh: Làm mẫu phần mềm kết hợp nhiều chức năng của sản phẩm cuối cùng nhưng bỏ qua những khía cạnh mà khách hàng không thấy được như cập nhật file hay xử lý lỗi.
- Bản mẫu nhanh phải được kiểm tra bởi khách hàng và người sử dụng.
- Bản mẫu nhanh có thể thay đổi cho đến khi khách hàng và người sử dụng cho rằng nó có những chức năng mà họ mong muốn.

Viết tài liệu pha yêu cầu:

- Tài liệu pha yêu cầu:
  - Có bản mẫu: Bản mẫu nhanh. Bản ghi thỏa thuận với khách hàng và người sử dụng về cơ sở xây dựng và sửa đổi bản mẫu.
  - Không có bản mẫu (nhóm quyết định không xây dựng bản mẫu): Mô tả nhu cầu khách hàng. Tài liệu này phải được khách hàng, người sử dụng và nhóm phát triển kiểm tra trước khi nhóm SQA xem xét.

2.4 PHA ĐẶC TẢ

- Khi khách hàng cho rằng nhóm phát triển hiểu được yêu cầu, nhóm đặc tả viết tài liệu đặc tả để mô tả chức năng của sản phẩm (những gì sản phẩm cần có + ràng buộc).

- Đặc tả bao gồm những input của sản phẩm và output được yêu cầu
- Ví dụ: Khách hàng cần tính bảng lương thì input bao gồm mức trả cho mỗi nhân viên, thông tin từ hồ sơ cá nhân để tính thuế… và output là số lương thuế, chi bảo hiểm,…
- Yêu cầu của đặc tả
    - Không nhập nhằng: không sử dụng thuật ngữ như tiện lợi, đầy đủ chức năng, nhanh, 98%...
    - Đầy đủ: Thể hiện mọi yêu cầu của khách hàng
    - Phi mâu thuẫn: không chứa mâu thuẫn
    - Theo dõi được (Traceability): có thể lần theo phán đoán trong đặc tả trở lại phán đoán đưa ra bởi khách hàng trong pha yêu cầu. Nếu đặc tả được trình bày đúng phương pháp, có đánh chỉ số,… thì nhóm SQA sẽ ít gặp khó khăn. Nếu có bản mẫu thì phán đoán liên quan của đặc tả phải theo dõi được đến bản mẫu.
- Một khi đặc tả được hoàn thành và đã thông qua thì hình thành kế hoạch quản lý quá trình sản xuất phần mềm (SPMP : The software product management plan).
- Yêu cầu kế hoạch:
    - SPMP cần nêu lên thời gian thực hiện, chi phí cho từng pha, gán trách nhiệm cá nhân cho từng pha, thời hạn hoàn thành cho mỗi pha.
    - Mô hình vòng đời nào sẽ sử dụng, cấu trúc tổ chức, kỹ thuật và CASE sử dụng, lịch tình, chi phí…

Kiểm thử pha đặc tả:

- Nguồn gốc chính của lỗi trong các phần mềm đã phân phối đến nay là những lỗi trong tài liệu đặc tả và những lỗi này chỉ được phát hiện khi tổ chức khách hàng sử dụng nó.
- Duyệt xét lại (Review) là cách tốt nhất để kiểm tra đặc tả. Mục đích là xác định đặc tả có đúng không. Nhóm SQA chủ trì cuộc họp với đại diện nhóm đặc tả và khách hàng.

2.5 PHA THIẾT KẾ

- Đặc tả - What
- Thiết kế - How
- Giữ lại quyết định thiết kế
    - Thời điểm kết thúc
    - Cho nhóm bảo trì
    - Thiết kế nên mở (open-ended)
- Thiết kế kiến trúc : Phân rã sản phẩm thành mô đun
- Thiết kế chi tiết: Thiết kế các mô đun: cấu trúc dữ liệu, thuật toán

Kiểm thử pha thiết kế

- Tài liệu viết sao cho dễ theo dõi
- Duyệt tài liệu

2.6 PHA CÀI ĐẶT

- Cài đặt thiết kế chi tiết thành chương trình

Kiểm thử pha cài đặt:

- Rà soát
- Các test case
  - Test không hình thức (desk checking)
  - Test hình thức (Formal testing) do nhóm SQA

2.7 TÍCH HỢP

- Kết hợp các mô đun và kiểm thử toàn bộ sản phẩm

Tài liệu pha tích hợp

- Mã nguồn có chú thích
- Các test cases

2.8 CẢI TIẾN TIẾN TRÌNH PHẦN MỀM

- Bắt đầu từ Bộ Quốc Phòng Mỹ
- Software Engineering Institue (SEI)
- Vấn đề:
  - Quản lý tiến trình phần mềm kém
- Cải tiến tiến trnfh phần mềm
  - Capability maturity model (CMM)
  - ISO 9000-Series
  - ISO/IEC 15504

CMM: Capability Maturity Model

- Không phải mô hình vòng đời
- Tập các chiến lược cải tiến tiến tình phần mềm
  - SW-CMM for software
  - P-CMM for human resource (“people”)
  - SE-CMM for systems engineering
  - IPD-CMM for integrated product development
  - SA-for software acquisition
- Các chiến lược được thống nhất thành CMMI (Capability maturity model integration)

SW-CMM

- A strategy for improving the software process
  - Put forward in 1986 by the SEI
  - Fundamental idea:
  - Improving the software process leads to
    - Improved software quality

        - Delivery on time, within budget
    - Improved management leads to
        - Improved techniques
- Five levels of "maturity" are defined
    - Organization advances stepwise from level to level

Level 1: Initial Level

- Ad hoc approach
    - Entire process is unpredictable
    - Management consists of responses to crises
- Most organizations world-wide are at level 1

Level 2: Repeatable Level

- Basic software management
    - Management decisions should be made on the basis of previous experience with similar products
    - Measurements ("metrics") are made
    - These can be used for making cost and duration predictions in the next project
    - Problems are identified, immediate corrective action is taken

Level 3: Defined Level

- The software process is fully documented
    - Managerial and technical aspects are clearly defined
    - Continual efforts are made to improve quality, productivity
    - Reviews are performed to improve software quality
    - CASE tools are applicable *now* (and not at levels 1 or 2)

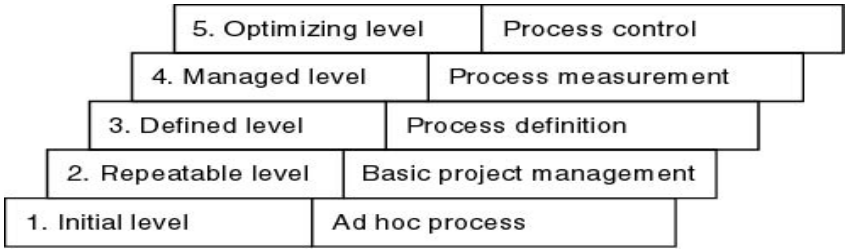
Level 4: Managed Level

- Quality and productivity goals are set for each project
    - Quality, productivity are continually monitored
    - Statistical quality controls are in place

Level 5: Optimizing level

- Continuous process improvement
    - Statistical quality and process controls
    - Feedback of knowledge from each project to the next

Kết luận:



Tiến trình chính

- Có các tiến trình chính (key Process Area) cho mỗi mức
- Mức 2:
  - Quản lý yêu cầu
  - Lập kế hoạch dự án
  - Theo dõi dự án
  - Quản lý cấu hình
  - Bảo đảm chất lượng
- So sánh
  - Mức 2: Phát hiện và sửa lỗi
  - Mức 3: Ngăn chặn lỗi

Kinh nghiệm

- Cần:
  - 3 đến 5 năm để từ mức 2 lên mức 3
  - 1.5 đến 3 năm từ mức 2 lên mức 3
- Original idea: Defense contracts would be awarded only to capable firms
- Profitability
  - Hughes Aircraft (Fullerton, CA) spent $500K (1987–90)
    - Savings: $2M per year moving from level 2 to level 3
  - Raytheon moved from level 1 in 1988 to level 3 in 1993
    - Productivity doubled
    - Return of $7.70 per dollar invested in process improvement

Other SPI Initiatives

- Other software process improvement (SPI) initiatives:
  - ISO 9000-series
  - ISO/IEC 15504

ISO 9000

- Set of five standards for industrial activities
  - ISO 9001 for quality systems
  - ISO 9000-3, guidelines to apply ISO 9001 to software
  - There is an overlap with CMM, but they are not identical
  - *Not* process improvement
  - Stress on documenting the process
  - Emphasis on measurement and metrics
  - ISO 9000 is required to do business with the E.U.
  - Also by many U.S. businesses, for example, GE
  - More and more U.S. businesses are ISO 9000 certified

ISO/IEC 15504

- Original name: Software Process Improvement Capability dEtermination (SPICE)
  - International process improvement initiative
  - Started by British Ministry of Defence (MOD)
  - Includes process improvement, software procurement
  - Extends and improves CMM, ISO 9000
  - Framework, not a method
    - CMM, ISO 9000 conform to this framework
  - Now referred to as ISO/IEC 15504
  - Or just 15504 for short

Process Improvement Data

Category	Range	Median	Number of Data Points
Years engaged in software process improvement (SPI)	1–9	3.5	24
Yearly cost of SPI per software engineer	$4[illegible]–$2004	$1375	5
Productivity gain per year	9%[illegible]7%	35%	4
Early defect detection gain per year	6%–2[illegible]	22%	3
Yearly reduction in time to market	15%–23%	19%	2
Yearly reduction in post-release defect re[illegible]ts	[illegible]0%–94%	39%	5
Business value (saving/cost of SPI)	[illegible].0–8.8:1	5.0:1	5

- SEI report on 13 org[illegible]nizatio[illegible]s in the original study
- They used a variety of p[illegible]ocess improvement techniques, not just SW–CMM

CMM Level	Number of Projects	[illegible]elative Decrease in Duration	Faults per MEASL Detected during Development	Relative Productivity
Level 1	3	1.0	—	—
Level 2	9	3.2	890	1.0
Level 3	5	2.7	411	0.8
Level 4	8	5.0	205	2.3
Level 5	9	7.8	126	2.8