Chương 2 : Vòng đời phần mềm

1, Hệ thống/ phần mềm

Hệ thống

- Một hệ thống bao gồm:

+, Phần cứng, hệ thống mạng, phần mềm\*, tài liệu

+, Phần mềm\* : bao gồm phần mềm nền tảng (như HĐH), phần mềm trung gian (middle software), và ứng dụng doanh nghiệp (Business Application Software)

2, Vòng đời phần mềm

- Là khoảng thời gian từ khi phần mềm được sinh ra cho đến khi chết đi (từ lúc hình thành đáp ứng yêu cầu, vận hành, bảo dưỡng cho đến khi loại bỏ không đâu dùng)

- Quy trình phần mềm (vòng đời phần mềm) được phân chia thành các pha chính : phân tích, thiết kế, chế tạo, kiểm thử, bảo trì. Biểu diễn các pha có thể khác nhau theo từng mô hình.

- Các pha trong vòng đời PM

Tất cả các sản phẩm phần mềm đều trải qua ít nhất các giai đoạn sau:

+, Yêu cầu – Xác định nhu cầu của khách hàng và các ràng buộc của sản phẩm

+, Thiết kế – Xác định cấu trúc/tổ chức của hệ thống phần mềm

+, Mã hoá – Viết phần mềm

+, Kiểm thử – Vận hành hệ thống để tìm và loại bỏ các khiếm khuyết

+, Bảo trì – Sửa chữa và nâng cao sản phẩm sau khi khách hàng triển khai

- Các mô hình vòng đời phần mềm

Quá trình là một tập hợp các hoạt động, với các đầu vào và đầu ra được xác định rõ ràng, để hoàn thành một số nhiệm vụ

Mô hình vòng đời là một mô tả về một quá trình thực hiện một sản phẩm phần mềm trong toàn bộ hoặc một phần vòng đời của nó

+, Các mô hình vòng đời có xu hướng tập trung vào các pha chính của chu kỳ và mối quan hệ của chúng với các pha khác.

+, Xem xét chi tiết nhiều khía cạnh của việc phát triển và bảo trì

+, Mô hình vòng đời là một mô tả quy trình phần mềm.

3, Quy trình phát triển phần mềm

4, Các mô hình quy trình phần mềm

4.1 Mô hình thác nước

- Là mô hình vòng đời lâu nhất

- Được gọi là thác nước vì nó thường được vẽ với một chuỗi các hoạt động qua các giai đoạn của vòng đời “xuống dốc” từ trái sang phải: phân tích, yêu cầu, đặc tả, thiết kế, cài đặc, kiểm thử, bảo trì.

Chương 3: Phương pháp Agile

1, Khái niệm

Phương pháp Agile:

+, Phản ứng hiệu quả (nhanh chóng và thích ứng) với sự thay đổi

+, Kết hợp quá trình tạo mẫu và dựa trên thử nghiệm

+, Có sự phát triển liên tục của lập trình

+, Liên tục xác nhận các yêu cầu của khách hàng

+, Mục tiêu: nhanh chóng phân phối phần mềm

2, Các nguyên lý cơ bản của phương pháp Agile

- Cách tốt hơn để phát triển phần mềm là làm và giúp người khác làm. Từ đó, các giá trị sau sẽ được nhận ra :

+, Các cá nhân và tương tác qua các quy trình và công cụ

+, Phần mềm làm việc dựa trên tài liệu toàn diện

+, Sự hợp tác của khách hàng trong quá trình đàm phán

+, Đáp ứng sự thay đổi so với việc tuân theo kế hoạch

- Các mục bên trái được coi trọng hơn bên phải

12 nguyên lý:

1, Ưu tiên cao nhất của là làm hài lòng khách hàng thông qua việc phân phối sớm và liên tục các phần mềm có giá trị

2, Hoan nghênh các yêu cầu thay đổi, ngay cả trong giai đoạn muộn của việc phát triển. Các quy trình nhanh khai thác sự thay đổi vì lợi thế cạnh tranh của khach hàng

3, Cung cấp sản phẩm phần mềm thường xuyên, từ vài tuần đến vài tháng, ưu tiên khoảng thời gian ngắn hơn.

4, Người kinh doanh và nhà phát triển phải làm việc cùng nhau hàng ngày trong suốt dự án

5, Xây dựng các dự án xung quanh những cá nhân có động lực. Cung cấp cho họ môi trường và sự hỗ trợ mà họ cần, và tin tưởng để học hoàn thành công việc

6, Phương pháp hiệu quả nhất để truyền tải thông tin đến và trong nhóm phát triển là trò chuyện trực tiếp

Ưu, nhược

- Phù hợp với những dự án nhỏ thường có những yêu cầu không được xác định rõ ràng (có thể thay đổi thường xuyên.)

- Khách hàng có thể được xem trước từng phần dự án trong quá trình phát triển, luôn sẵn sàng cho bất kỳ thay đổi từ phía khách hàng.

- Agile chia dự án thành những phần nhỏ và giao cho nhóm phát triển, hàng ngày tất cả mọi người phải họp giải quyết những vấn đề nảy sinh nếu có nhằm đảm bảo đúng quy trình phát triển dự án.

- Tỉ lệ thành công của các dự án sử dụng Agile thường cao hơn các quy trình

- Khó xác định loại dự án phần mềm nào phù hợp nhất cho cách tiếp cận Agile

- Nhiều tổ chức lớn gặp khó khan trong việc chuyển từ phương pháp truyền thống sang một phương pháp Agile (linh hoạt)

- Khi Agile có rủi ro :

-> Phát triển quy mô lớn (> 20 nhà phát triển)

-> Phát triển phân tán (các nhóm không nằm chung)

-> Khách hàng hoặc người liên hệ không đáng tin cậy

-> Bắt buộc quy trình nhanh trong nhóm phát triển

-> Nhà phát triển thiếu kinh nghiệm

4, Extreme Programming

- Không thể lập trình cho đến khi biết mình đang làm gì

- Cần khắc phục ở giai đoạn đầu tiên khi hệ thống đi vào sản xuất càng nhanh càng tốt. Tuy nhiên, mọi dự án đều phải bắt đầu từ điểm nào đó

- Kết hợp phân tích tổng thể dưới dạng các vấn đề. Mỗi vấn đề phải theo định hướng kinh doanh, có thể kiểm tra và ước tính được

- Một tháng là một khoảng thời gian dài để đưa ra nữhng vấn đề cho dự án

- Xác định thời điểm lập trình

- Để XP thành công, cần có kiến thức chuyên môn

- Biến thành mã thử nghiệm

- Kiểm thử đơn vị là trọng tâm của XP, trong đó hai điểm xoay qunah các chiến lược kiểm thử thông thường giúp kiểm tra hiệu quả hơn

- Các lập trình viên viết các test riêng và cần viết trước khi viết mã.

- Thiết kế phát triển – Không được phá vỡ các thử nghiệm hiện có và cũng phải hỗ trợ phát triển gia tăng có thể mở rộng.

Ưu điểm:

- Nếu được hoàn thành tốt, XP cải thiện tình đồng đội

- Xây dựng năng lực thực sự ở tất cả các thành viên trong nhóm

- Làm cho công việc thú vị

- TDD (Test-driven development) hướng dẫn các nhà phát triển về cách viết mã chất lượng và cách cải thiện quan niệm về thiết kế; giúp cải thiện việc ước lượng.

- Cung cấp nhiều công cụ cho quản lý, bao gồm khả năng dự đoán, tính linh hoạt của các nguồn lực, tính nhất quán và khả năng hiển thị về những gì thực sự đang diễn ra

- Cung cấp cho khách hàng khả năng xem xét công ty có thể thực hiện công việc của mình hay không

- Không lãng phí thời gian và không tạo ra nhiều tài liệu vô ích.

Nhược điểm:

- Thiết kế trở nên tiềm ẩn thay vì rõ ràng

5, Scrum

- Trong phát triển phần mềm, công việc là đưa ra một bản phát hành.

- Nhiệm vụ: tăng tốc việc phát hành sản phẩm

- Quá trình phát triển được chia thành một loạt các lần lặp được gọi là sprint

- Trước mỗi sprint, các thành viên trong nhóm xác định các công việc còn tồn

- Khi kết thúc sprint, nhóm sẽ xem xét để nêu rõ các kinh nghiệm và kiểm tra tiến độ

- Các khái niệm chính :

+, Burndown chart là biểu đồ cho thấy công việc còn tồn trong sprint (được cập nhật mỗi ngày) sử dụng để theo dõi tiến trình và quyết định khi các mục phải được loại bỏ khỏi sprint backlog và hoãn lại sprint tiếp theo.

+, Product backlog là danh sách đầy đủ cac yêu cầu – bao gồm lỗi, yêu cầu nâng cao, cải tiến khả năng sử dụng và hiệu suất – hiện không có trong bản phát hành sản phẩm

+, ScrumMaster là người chịu trách nhiệm quản lý dự án

Sprint backlog là danh sách các công việc được chỉ định cho một sprint, nhưng chưa hoàn thành

Chương 6: Kỹ nghệ yêu cầu phần mềm

1. Khái niệm

- Các đặc tính của hệ thống hay sản phẩm do khách hàng – người sử dụng PM – đặt ra => Xác định được phần mềm đáp ứng được các yêu cầu và mong muốn của khách hàng – người sử dụng phần mềm.

- Khởi đầu (Inception) Hỏi một loạt các câu hỏi để xác định:

+, Hiểu biết căn bản về vấn đề giải quyết

+, Người đang cần giải pháp

+, Loại giải pháp mong muốn

+, Mức độ hiệu quả ban đầu của việc trao đổi thông tin giữa khách hàng và nhà phát triển

- Khám phá (Elicitation) : tìm ra yêu cầu của tất cả khsach hàng

- Xây dựng: tạo ra mô hình phân tích xác định dữ liệu, chức năng và hành vi được yêu cầu

- Đàm phán: đồng ý với một hệ thóng có thể bàn giao một cách thực tế đối với cả 2 bên

- Đặc tả (Specification): có thể là một/ nhiều những thứ sau :

+, Một tài liệu được viết

+, Một tập hợp các mô hình

+, Một hình thức biểu diễn toán học

+, Một tập các kịch bản người dung (use-case)

+, Một nguyên mẫu

- Đánh giá (Validation): tạo cơ chế xem xét các vấn đề:

+, Sai sót trong nội dung hoặc giải thích

+, Phần được yêu cầu làm rõ

+, Thông tin bị thiếu

+, Mâu thuẫn

+, Yêu cầu không thực tế, không thể đạt được

- Quản lý các yêu cầu (Requirements management)

1. Tầm quan trọng của yêu cầu phần mềm

- Nguồn gốc yêu cầu phần mềm:

Người sử dụng (khách hàng): theo mô hình phân lớp của yêu cầu phần mềm Khách hàng được chia làm hai loại:

+, Khách hàng cung cấp business requirêmnt: cung cấp các thông tin về công ty, về các đặc điểm ở mức độ cao, về mô hình và phạm vi của hệ thống

+, Khách hàng cung cấp user requirement: cung cấp các thông tin về từng nhiệm vụ cụ thể mà họ sẽ làm việc với phần mềm

Cần phải phối hợp, kết hợp chặt chẽ với hai phân loại khách hàng trên

- Đặc điểm của khách hàng phần mềm

Khách hàng chỉ có những ý tưởng còn mơ hồ về phần mềm cần phải xây dựng để phục vụ công việc của họ

Cho nên chúng ta phải sẵn sàng, kiên trì theo đuổi để đi từ các ý tưởng mơ hồ đó đến “Phần mềm có đầy đủ các tính năng cần thiết”

Khách hàng rất hay thay đổi các đòi hỏi của mình, chúng ta nắm bắt được các thay đổi đó và sửa đổi các mô tả một cách hợp lý.