**Câu 1: 4đ**

1. **Cho biết 3 thách thức của công nghệ phần mềm**

Vấn đề thời gian: Một hệ thống bài bản cần thực hiện qua nhiều bước, tiêu tốn nhiều thời gian. Thậm chí để nhìn thấy được hiệu quả triển khai, doanh nghiệp có thể mất từ 2-5 năm.

Vấn đề chi phí: Chi phí triển khai của dự có thể lớn hơn nhiều so với chi phí ban đầu dự tính. Bởi trong quá trình triển khai, doanh nghiệp thường sẽ tiến hành tùy chỉnh. Càng tùy chỉnh nhiều, chi phí sẽ càng cao.

Vấn đề lỗi trước khi chuyển giao: ???

Vấn đề đo lường: ???

Chi phí bảo trì: Nếu lơ là bảo trì, doanh nghiệp có thể vô tình khiến hệ thống bị phá vỡ, giảm hiệu quả hoạt động, gia tăng thêm nhiều khoản chi phí phát sinh.

Bảo mật: Vấn đề bảo mật luôn là một trong những thách thức lớn nhất của công nghệ phần mềm. Các hacker và phần mềm độc hại luôn tìm cách xâm nhập vào các hệ thống và gây ra các vấn đề bảo mật.

Hiệu suất: Một trong những yêu cầu quan trọng của phần mềm là hiệu suất. Với các ứng dụng phần mềm lớn và phức tạp, việc đảm bảo hiệu suất đúng mức là một thách thức lớn đối với các nhà phát triển.

1. **Tính hiệu quả của phần mềm, là một trong 4 thuộc tính thiết yếu để đánh giá một phần mềm, có nghĩa là gì?**

Khả năng bảo trì (Maintainability): phần mềm có thể duy trì hoạt động, có thể điều chỉnh và mở rộng để thoả mãn những yêu cầu luôn thay đổi.

Mức độ tin cậy (Reliability-Dependability): phần mềm phải được tin cậy, bảo mật và chính xác.

Không nên gây ra thiệt hại vật chất hoặc kinh tế trong trường hợp lỗi hệ thống. Người dùng độc hại sẽ không thể truy cập hoặc làm hỏng hệ thống

*Hiệu quả (efficiency): phần mềm không nên sử dụng lãng phí tài nguyên của hệ thống như thời gian, bộ nhớ, CPU, mạng và bất kỳ nguồn tài nguyên nào khác được sử dụng trong quá trình chạy phần mềm. (khả năng đáp ứng, thời gian xử lý, sử dụng bộ nhớ)*

*Một phần mềm được coi là hiệu quả nếu nó có khả năng hoàn thành công việc của nó một cách nhanh chóng và đáp ứng các yêu cầu hiệu suất với mức tài nguyên được sử dụng là hợp lý. Tính hiệu quả cũng liên quan đến khả năng của phần mềm hoạt động trơn tru và ổn định trong thời gian dài và đáp ứng được nhu cầu của người sử dụng.*

Khả năng được chấp nhận (acceptability-Usability): người sử dụng phải chấp nhận phần mềm. Điều đó có nghĩa là nó phải dễ hiểu, sử dụng được và tương thích với các hệ thống khác

1. **Trong một bộ yêu cầu tốt thì có thể có các yêu cầu tranh chấp với nhau. Cho 1 ví dụ về 2 yêu cầu tranh chấp.**

*Có 3 loại yêu cầu:*

*-Yêu cầu chức năng: chức năng dịch vụ hệ thống cung cấp*

*-Yêu cầu phi chức năng: những ràng buộc về tiêu chuẩn, thời gian, qui trình phát triển…, chủ yếu là những yêu cầu về chất lượng.*

*+Các yêu cầu về sản phẩm: hiệu năng, độ tin cậy…*

*+Các yêu cầu của tổ chức (khách hàng hay người phát triển): thời gian bàn giao, yêu cầu phù hợp với hệ thống cũ…*

*+Các yêu cầu ngoài: được xác định từ các tác nhân từ bên ngoài như các yêu cầu về luật pháp, yêu cầu tôn trọng tính riêng tư, tương tác với hệ thống bên ngoài…*

*-Yêu cầu miền ứng dụng: phản ảnh những đặc trưng của miền ứng dụng*

*VD yêu cầu phi chức năng:*

*YÊU CẦU CỦA SẢN PHẨM MHC-PMS sẽ có sẵn cho tất cả các phòng khám trong giờ làm việc bình thường (Thứ Hai – Thứ Sáu, 08.30–17.30). Thời gian ngừng hoạt động trong giờ làm việc bình thường không được vượt quá năm giây trong một ngày bất kỳ.*

*YÊU CẦU CỦA TỔ CHỨC Người sử dụng hệ thống MHC-PMS sẽ tự xác thực bằng chứng minh thư của cơ quan y tế.*

*YÊU CẦU BÊN NGOÀI Hệ thống sẽ thực hiện các điều khoản về quyền riêng tư của bệnh nhân như được nêu trong HStan-03-2006-priv*

**1 ví dụ về 2 yêu cầu tranh chấp:**

Yêu cầu phải hoạt động trên cả hai hệ điều hành Windows và MacOS và yêu cầu phải tuân theo chuẩn định dạng tệp tin DOCX. Nhưng đối với một số tính năng đặc biệt, định dạng tệp tin DOCX có thể không được hỗ trợ trên hệ điều hành MacOS, dẫn đến mâu thuẫn giữa yêu cầu phải hoạt động trên cả hai hệ điều hành và yêu cầu phải sử dụng định dạng tệp tin DOCX.

Yêu cầu phải có khả năng tương thích ngược với các phiên bản phần mềm cũ hơn, trong khi đó yêu cầu khác yêu cầu phải có khả năng tương thích với các phiên bản phần mềm mới nhất. Tuy nhiên, để tương thích ngược với các phiên bản cũ, phải sử dụng các tính năng đã bị thay đổi hoặc bị gỡ bỏ trong phiên bản mới nhất, dẫn đến mâu thuẫn giữa yêu cầu tương thích với các phiên bản cũ và yêu cầu tương thích với các phiên bản mới nhất.

1. **Kể ra 2 trường hợp làm cho một yêu cầu là không khả thi.**

*Phân tích khả thi cho biết hệ thống với những yêu cầu xác định có thể thực hiện trong những điều kiện về kỹ thuật, tài nguyên, ngân sách… Một số vấn đề:*

*Hệ thống có đóng góp vào mục tiêu của tổ chức hay không?*

*Hệ thống có thể được xây dựng bằng cách sử dụng công nghệ hiện tại, trong tiến độ và ngân sách cho phép?*

*Hệ thống có được tích hợp với các hệ thống khác đang sử dụng hay không?...*

*Những câu hỏi đặt ra để phân tích khả thi:*

*Vấn đề xử lý hiện tại như thế nào?*

*Hệ thống đề nghị cung cấp những tiện ích gì?*

*Nếu hệ thống không được cài đặt thì sao?*

*Hệ thống đề xuất sẽ trợ giúp nghiệp vụ theo cách thức nào?*

*Những rắc rối về việc tích hợp là gì?*

*Có cần kỹ thuật mới không?*

*Cần có những kỹ năng gì để thực hiện và sử dụng?*

**2 trường hợp yêu cầu không khả thi**

Một số trường hợp yêu cầu không khả thi trong phần mềm bao gồm:

Yêu cầu không khả thi về mặt kỹ thuật: Một yêu cầu có thể bị coi là không khả thi nếu nó yêu cầu tính năng hoặc chức năng nằm ngoài khả năng của phần mềm hoặc yêu cầu phải sử dụng công nghệ không có sẵn trong phần mềm.

Yêu cầu không khả thi về mặt thời gian và ngân sách: Nếu yêu cầu yêu cầu một lượng công việc lớn hơn hoặc chi phí cao hơn so với những gì được dự đoán ban đầu hoặc ngân sách không cho phép, yêu cầu đó có thể được coi là không khả thi.

Ví dụ:

Yêu cầu phải có tính năng xử lý video trên một máy tính xách tay thông thường với độ phân giải cao và tốc độ khung hình nhanh. Điều này có thể không khả thi vì yêu cầu một cấu hình phần cứng cao hơn và chưa được hỗ trợ bởi phần mềm.

Yêu cầu phải có một tính năng độc đáo và phức tạp, yêu cầu một lượng công việc lớn và chi phí cao hơn so với dự đoán ban đầu của dự án phần mềm, vì vậy yêu cầu này có thể không khả thi trong ngân sách và thời gian cho phần mềm.

1. **Cho biết 2 nhược điểm và 2 ưu điểm trong số các ưu nhược điểm của quy trình Thác nước.**

|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| Dễ triển khai quy trình dễ hiểu | Không thể thay đổi yêu cầu |
| Hoạt động hiệu quả cho kết quả tốt với dự án nhỏ | Không thích hợp với dự án dài và phức tạp |
| Chất lượng mỗi phần và cả dự án rất chặt chẽ | Khó di chuyển qua lại giữa các pha |
| Yêu cầu đầu vào rõ ràng thì tiến trình làm việc dễ dàng, chất lượng | Rủi ro cao nếu không phát hiện từ sớm. |
| Kết quả ghi chép dễ dàng |  |

1. **Cho biết 2 khác biệt nền tảng của phương pháp PTPMLH (Alige) và phương pháp cũ?**

Khác biệt dễ nhận thấy của PPPTPMLH đó là lượng giấy tờ tài liệu ít hơn và có thể nói là tập trung vào việc lập trình hơn. Nhưng ẩn đằng sau đó là hai khác biệt nền tảng quan trọng: “thích ứng thay vì dự đoán” và “hướng đến con người thay vì qui trình”.

PPPTPMLH đề cao tính chủ động và sáng tạo của các cá nhân tham gia, và đặc biệt là việc trao đổi thông tin giữa các thành viên.

PPPTPMLH không khước từ sự tổ chức nhưng nó cố gắng cân bằng giữa sự tổ chức và sự linh hoạt, cân bằng giữa việc không có qui trình nào cả và qui trình quá chi li và cứng nhắc

1. **Giải thích ngắn lập trình cặp đôi (pair programing)**

là kiểu lập trình đòi hỏi hai [kỹ sư phần mềm](https://vi.wikipedia.org/wiki/K%E1%BB%B9_s%C6%B0_ph%E1%BA%A7n_m%E1%BB%81m) cùng tham gia một nỗ lực lập trình chung trên một máy trạm, nghĩa là chỉ có một màn hình, một bàn phím. Mỗi người thực hiện việc mà người kia hiện không làm. Trong đó, một người có vai trò là coding(gọi là Driver), một người có vai trò theo dõi, hướn dẫn hay định hướng hỗ trợ(Driver), phát hiện lỗi. Vai trò này luôn thay phiên nhau trong 1 khoảng thời gian hợp lý

1. **Phần mềm đầy đủ hoàn tất trong 3 tháng nhưng trong tháng đầu tiên đã có sản phẩm chuyển giao cho khách hàng, như vậy khả năng cao là đã qáp dụng quy trình nào, theo quy trình này thì yêu cầu có cần phải xác định hoàn toàn đầy đủ ngay từ đầu không?**

Việc xác định hoàn toàn đầy đủ từ đầu là rất quan trọng trong quá trình phát triển phần mềm để đảm bảo sản phẩm cuối cùng đáp ứng được các yêu cầu của khách hàng. Tuy nhiên, nếu bạn đã chuyển giao sản phẩm cho khách hàng trong tháng đầu tiên, có thể hiểu rằng đội ngũ phát triển phần mềm của bạn đã áp dụng quy trình Agile hoặc Scrum để phát triển phần mềm.

Trong Agile và Scrum, đội phát triển phần mềm sẽ làm việc theo các chu kỳ phát triển ngắn (sprint), trong đó, các yêu cầu sẽ được phân tích và xác định một cách chi tiết hơn để đảm bảo sản phẩm đáp ứng được các yêu cầu của khách hàng. Tuy nhiên, không phải tất cả các yêu cầu đều cần được xác định đầy đủ từ đầu.

Trong quy trình Agile và Scrum, yêu cầu có thể được xác định và điều chỉnh dần theo từng chu kỳ phát triển, và sản phẩm sẽ được chuyển giao cho khách hàng ở cuối mỗi chu kỳ phát triển. Việc này giúp đảm bảo rằng sản phẩm luôn đáp ứng được yêu cầu của khách hàng và có thể được điều chỉnh một cách nhanh chóng nếu cần thiết.

Vì vậy, việc xác định hoàn toàn đầy đủ từ đầu hay không phụ thuộc vào quy trình phát triển phần mềm mà đội ngũ phát triển đang áp dụng. Nếu đội ngũ của bạn áp dụng Agile hoặc Scrum, việc chuyển giao sản phẩm cho khách hàng trong tháng đầu tiên là có thể và hợp lý.

**Câu 2: Vẽ biểu đồ luồng dữ liệu DFD cấp 2**

**THÔNG TIN NGOÀI LỀ**

Ba loại yêu cầu

Yêu cầu thông thường (Normal Requirements): không có thì không đáp ứng yêu cầu

Yêu cầu mong đợi (Expected Requirements): có thì sản phẩm tốt hơn

Yêu cầu kỳ thú (Exciting Requirements): ngay cả khách hàng cũng nghĩ đến

Cấp độ đặc tả yêu cầu

Yêu cầu nghiệp vụ:

Yêu cầu người dùng:

Viết chủ yếu cho người dùng

Thường bằng ngôn ngữ tự nhiên cộng với các biểu đồ

Mô tả các dịch vụ và những ràng buộc hoạt động

Yêu cầu hệ thống

Tài liệu có cấu trúc mô tả chi tiết chức năng, dịch vụ và ràng buộc

Có thể là một phần của hợp đồng, xác định những gì cần phải thực hiện

Một số quy trình cơ bản xây dựng phần mềm

**2.2.1 Mô hình thác nước**

Phát triển trình tự các bước mỗi giai đoạn xác định chuẩn vào và ra mô hình dễ hiểu, dễ thực hiện.

Cung cấp một cấu trúc rõ ràng cho những nhân viên thiếu kinh nghiệm hay yếu về kỹ thuật.

Việc chuyển từ giai đoạn này sang giai đoạn kế được thực hiện khi thỏ một kiểm tra xác định đồng thuận giữa thành viên dự án và khách hàng.

Áp dụng cho phần mềm chất lượng cao, yêu cầu chất lượng nổi trội hơn, yêu cầu lịch biểu và chi phí yêu cầu cố định không thay đổi thường xuyên tài nguyên phong phú.

|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| Dễ triển khai quy trình dễ hiểu | Không thể thay đổi yêu cầu |
| Hoạt động hiệu quả cho kết quả tốt với dự án nhỏ | Không thích hợp với dự án dài và phức tạp |
| Chất lượng mỗi phần và cả dự án rất chặt chẽ | Khó di chuyển qua lại giữa các pha |
| Yêu cầu đầu vào rõ ràng thì tiến trình làm việc dễ dàng, chất lượng | Rủi ro cao nếu không phát hiện từ sớm. |
| Kết quả ghi chép dễ dàng |  |

**2.2.2 Mô hình phát triển gia tăng**

Với phương pháp chia nhỏ thành từng bản khác nhau từ một chu kỳ lớn sẽ được phân chia thành nhiều chu kỳ nhỏ ứng với từng bảng và do đó ta có được một đa chu kỳ phát triển.

Mỗi chu kỳ nhỏ ứng với một bản phân chi gọi là modul đơn giản. Và dễ dàng quản lý hơn. Mỗi modul cũng được xác định theo từng bước như là phân tích đọc, yêu cầu dự án thiết kế, tiến hành coggins và thực hiện tết.

Nguyên lý như việc xếp một bức tranh từ các miếng ghép miếng ghép nào được hoàn thành trước thì sẽ cho ta một phần bức tranh được thể hiện trước theo thời gian, số miếng ghép được gia tăng và sản phẩm ngày càng hoàn thiện.

|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| * Sớm tạo ra nguyên mẫu của sản phẩm trong vòng đời phát triển của nó. * Độ linh hoạt cao khi thay đổi yêu cầu sẽ ít chi phí hơn * Chia nhỏ thì dễ test, dodul đơn giản thì test đơn giản * Sớm kết thúc giảm chi phí cho lần đầu giao sản phẩm * Dễ quản lý chi phí phát sinh * Thực hiện nhiều bước đồng thời. | * Phải có khả năng thiết kế tốt, phương pháp tốt, hiểu rõ yêu cầu và biết cách phân chia cho hợp lý * Chi phí cao hơn thác nước * Cần xác định rõ các giao tiếp giữa các modul mà thời gian hoàn thành cách biệt nhiều. |

Áp dụng khi:

* Yêu cầu rõ ràng, đầy đủ, mong muốn tiến hóa dần của sản phẩm
* Khi cần nhanh chóng đưa sản phẩm với chức năng cơ bản ra thị trường sớm
* Khi có 1 tính năng hay mục tiêu có nguy cơ lỗi cao

**2.2.3 Mô hình xoắn ốc**

Mỗi vòng lặp đều có phân tích rủi ro, chỉ báo sớm những rủi ro không thể khắc phục với phí tổn không cao.

Quy trình này áp dụng cho những dự án cần nhiều thời gian và có rủi ro từ trung bình đến cao.

Có thể được xem là sự kết hợp giữa mô hình thác nước (Waterfall model) và mô hình mẫu (Prototype model) và đồng thời thêm phân tích rủi ro (Risk assessment).

|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| * Phân tích rủi ro dự án được đầy lên làm một phần thiết yếu trong quy trình xoắn ốc để tăng độ tin cậy của dự án. * Xây dựng dự án có sự kết hợp các mô hình khác vào phát triển (Thác nứơc, mô hình mẫu…) * Cho phép thay đổi tuỳ theo yêu cầu cho mỗi vòng xoắn ốc. * Nó được xem như là một mô hình tổng hợp của các mô hình khác. * Không chỉ áp dụng cho phần mềm mà còn phải cho cả phần cứng. * Một rủi ro nào đó không được giải quyết thì chấm dứt dự án. * Các vòng tròn được lặp để đáp ưng được những thay đổi của người dùng * Kiểm soát rủi ro ở từng giai đoạn phát triển. * Đánh giá tri phí chính xác hơn các phương pháp khác | * Nó cần các kiến thức đánh giá rủi ro chuyên sâu. Việc đánh giá rủi ro tốn nhiều chi phí, không không thích hợp cho những dự án rủi ro thấp hay nhỏ * Mô hình phức tạp, khó sử dụng * Khó quản lý tiến trình và thuyết phục khách hàng * hức tạp và không thích hợp với các dự án nhỏ và ít rủi ro. * Cần có kỹ năng tốt về phân tích rủi ro. * Yêu cầu thay đổi thường xuyên dẫn đến lặp vô hạn * Chưa được dùng rộng dãi như mô hình thác nước hay là mẫu. * Đòi hỏi năng lực quản lý |

2.3 Các quy trình khác

2.3.1 Quy trình RUP

Giai đoạn 1 (Inception): khởi đầu

* Xác định phạm vi dự án, yêu cầu người dùng và các ràng buộc
* Xác định Yêu cầu nghiệp vụ, phân tích rủi ro, lập kế hoạch dự án (phân công, chi phí)
* Thiết kế kiến trúc hệ thống (quan tâm đến chi phí, lịch biểu, tài nguyên)
* Cấu hình môi trường làm việc, công cụ

Giai đoạn 2 (Elaboration): Hình thành

* Tinh chỉnh tài liệu ν Hoạch định những bước lặp
* Kế hoạch phát triển: qui trình, công cụ CASE
* Tinh chỉnh kiến trúc và chọn thành phần (component)

Giai đoạn 3 (Construction): Xây dựng

* Quản lý tiến trình tạo sản phẩm: tăng năng suất, đảm bảo chất lượng
* Tạo sản phẩm (alpha, beta, các phiên bản test khác)
* Kế hoạch triển khai ứng dụng: chuẩn bị phần mềm, huấn luyện người sử dụng, các biện pháp hỗ trợ…

Giai đoạn 4 (Transition): Chuyển giao

* Tạo sản phẩm xuất xưởng
* Kiểm tra sản phẩm, thu thập thông tin phản hồi

**2.3.2 Phương pháp phát triển phần mềm linh hoạt (PTPMLH - Agile software development)**

Thay vì chỉnh sửa toàn bộ sản phẩm trong 1 lần, các đội agile chia nó thành các phần khác nhau để kiểm tra và nâng cao riêng trước khi chuyển sang chu trình tiếp theo. Đảm bảo mỗi phần của phần mềm được lý tưởng hóa, phát triền và nhận được phản hồi theo thời gian thực từ khách hàng

Phương pháp Agile sơ khai được tạo ra để phục vụ cho ngành công nghiệp phát triển phần mềm và cải tiến sản xuất. Nó lỗ tác dụng giúp các nhà nghiên cứu phát hiện, điều chỉnh các vấn đề sai sót và khiếm khuyết một cách dễ dàng

|  |  |
| --- | --- |
| Ưu điểm | Nhược điểm |
| Phương pháp Agile sơ khai được tạo ra để phục vụ cho ngành công nghiệp phát triển phần mềm và cải tiến sản xuất. Nó lỗ tác dụng giúp các nhà nghiên cứu phát hiện, điều chỉnh các vấn đề sai sót và khiếm khuyết một cách dễ dàng  • Agile thay thế cho phương pháp truyền thống Waterfall, nó quản lý các nhóm làm việc chặt chẽ hơn và đẩy nhanh tiến độ sản xuất một sản phẩm hơn  • Thực hiện các thay đổi nhanh gọn. Mỗi dự án được chia thành nhiều giai đoạn làm việc và độc lập xử lý thông tin, vậy nên mọi thay đổi đều được thích nghi nhanh chóng  Thông tin cập nhật theo thời gian. Agile phù hợp với những dự án còn đang trong quá trình xây dựng kế hoạch, chưa có hết mọi mục tiêu dài hạn.  • Bàn giao nhanh hơn. Sau mỗi giai đoạn được hoàn thành là một phần của dự án được bàn giao  • Phản hồi của khách hàng và người dùng là kênh thông tin hữu ích Lắng nghe ý kiến từ mọi người giúp nâng cao chất lượng sản phẩm  . Cải tiến liên tục. Khi bạn tìm hiểu về mô hình Agile là gì, bạn biết được Agile luôn khuyến khích sự thay đổi và cập nhật nhanh chóng ở mọi giai đoạn | Khó lên kế hoạch dự án Mỗi dự án trong mô hình Agile sẽ có các giai đoạn riêng biệt ứng với thời gian hoàn thành khác nhau vì vậy rất khó để xác định khi nào là lúc sản phẩm hoàn thiện.  . Hướng dẫn và đào tạo chi tiết từng giai đoạn. Hiểu được phương pháp Agile là gì sẽ biết được nó phức tạp hơn nhiều so với phương pháp Waterfall. Các nhóm sẽ cần chuyên gia đào tạo để nắm bắt được mọi thông tin sản phẩm  ít tài liệu hướng dẫn. Khi học về mô hình Agile là gì chúng ta thường có rất ít tư liệu đễ hiểu các giai đoạn vì mỗi dự án có một đặc thủ riêng biệt  • Bắt buộc phải hợp tác để dự án thành công. Việc bắt buộc hợp tác trong phương pháp Agile là gì - chính là cần đổi bên phải giao tiếp trực tiếp để tìm ra phương án tốt nhất  • Chi phí cao Các chi phí của mô hình Agile là gì? Là những khoản chi cho việc đào tạo nhân sự cho từng giai đoạn nên sẽ tốn kém hơn phương pháp truyền thống |

•