**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

Bài tập cá nhân nhóm 11

Sinh viên thực hiện:

Nguyễn Văn Hiền



**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

Tuần 3

Part 1: Câu hỏi trắc nghiệm

**Câu hỏi 1: Pha nào trong mô hình lý thuyết vòng đời phát triển phần mềm chịu trách nhiệm chuyển đổi yêu cầu thành đặc tả kỹ thuật?**

* **Đáp án đúng: C. Pha phân tích**
* **Giải thích:** Pha phân tích là giai đoạn quan trọng trong vòng đời phát triển phần mềm, nơi các yêu cầu của khách hàng hoặc người dùng được xem xét và chuyển đổi thành tài liệu đặc tả kỹ thuật. Đặc tả kỹ thuật này giúp lập trình viên và các bên liên quan hiểu rõ hệ thống cần phát triển.

**Câu hỏi 2: Mô hình vòng đời nào phát triển phần mềm bằng cách tạo các phiên bản nhỏ và tăng dần tính năng?**

* **Đáp án đúng: B. Mô hình lặp và tăng trưởng**
* **Giải thích:** Mô hình **lặp và tăng trưởng (Incremental and Iterative Model)** cho phép phát triển phần mềm theo từng phiên bản nhỏ, mỗi phiên bản cải thiện hoặc bổ sung tính năng mới. Điều này giúp phần mềm linh hoạt hơn, có thể thích nghi với các thay đổi nhanh chóng trong yêu cầu của người dùng.

**Câu hỏi 3: Pha bảo trì trong vòng đời phát triển phần mềm bao gồm hoạt động nào?**

* **Đáp án đúng: B. Sửa lỗi và cập nhật tính năng mới**
* **Giải thích:** Giai đoạn bảo trì trong vòng đời phần mềm bao gồm việc **sửa lỗi (bug fixing)**, **tối ưu hiệu suất**, **nâng cấp tính năng** và **cập nhật bảo mật**. Viết mã nguồn (A) không thuộc giai đoạn bảo trì, mà thuộc giai đoạn phát triển. Gỡ bỏ phần mềm (C) liên quan đến giai đoạn ngừng hỗ trợ chứ không phải bảo trì.

**Câu hỏi 4: Mô hình thác nước phù hợp nhất với loại dự án nào?**

* **Đáp án đúng: B. Dự án có yêu cầu rõ ràng và ít thay đổi**
* **Giải thích:** Mô hình **thác nước (Waterfall Model)** là một phương pháp phát triển phần mềm tuyến tính, trong đó mỗi pha phải hoàn thành trước khi chuyển sang pha tiếp theo. Do đó, nó phù hợp với các dự án **có yêu cầu rõ ràng ngay từ đầu và ít thay đổi**. Nếu yêu cầu thay đổi nhiều (C, D), mô hình thác nước sẽ không linh hoạt bằng các mô hình khác như Agile hay mô hình xoắn ốc.

**Câu hỏi 5: Trong mô hình xoắn ốc, mỗi vòng xoắn tương ứng với:**

* **Đáp án đúng: B. Một chu kỳ lặp của toàn bộ quy trình phát triển**
* **Giải thích:** Mô hình **xoắn ốc (Spiral Model)** kết hợp các yếu tố của mô hình thác nước và mô hình lặp. Mỗi vòng xoắn tương ứng với một chu kỳ phát triển, bao gồm **phân tích rủi ro, thiết kế, triển khai, kiểm thử và đánh giá**. Việc lặp lại liên tục giúp cải thiện sản phẩm theo từng giai đoạn, đảm bảo phần mềm đáp ứng tốt nhất nhu cầu thực tế.

**Câu hỏi 6: Điểm yếu lớn nhất của mô hình xây và sửa là gì?**

* **Đáp án đúng: C. Khó kiểm soát chất lượng**
* **Giải thích:** Mô hình **xây và sửa (Build and Fix)** là phương pháp phát triển phần mềm đơn giản nhất, trong đó phần mềm được xây dựng và chỉnh sửa ngay khi có lỗi mà không có kế hoạch rõ ràng. Điều này dẫn đến:
  + Khó kiểm soát chất lượng sản phẩm do thiếu quy trình kiểm thử bài bản.
  + Code dễ bị lỗi và khó bảo trì.
  + Không phù hợp với các dự án lớn, yêu cầu độ tin cậy cao.

**Câu hỏi 7: Mô hình nào tập trung vào việc tạo các nguyên mẫu nhanh để thu thập phản hồi từ khách hàng?**

* **Đáp án đúng: B. Mô hình bản mẫu nhanh (Prototyping Model)**
* **Giải thích:** Mô hình **bản mẫu nhanh** cho phép phát triển nhanh một phiên bản thử nghiệm của phần mềm để khách hàng đánh giá và phản hồi. Điều này giúp giảm rủi ro và đảm bảo rằng sản phẩm cuối cùng đáp ứng nhu cầu thực tế của người dùng.

**Câu hỏi 8: Pha nào kết thúc vòng đời phát triển phần mềm?**

* **Đáp án đúng: C. Pha giải thể**
* **Giải thích:** Vòng đời phát triển phần mềm không chỉ dừng lại ở **triển khai và bảo trì**, mà còn bao gồm **giai đoạn giải thể** khi phần mềm không còn được sử dụng. Trong pha này, hệ thống có thể bị gỡ bỏ, dữ liệu được sao lưu, và tài nguyên phần cứng có thể được tái sử dụng cho mục đích khác.

**Câu hỏi 9: Điểm khác biệt chính giữa mô hình lặp và tăng trưởng với mô hình thác nước là gì?**

* **Đáp án đúng: B. Mô hình lặp và tăng trưởng phát triển theo từng đợt nhỏ**
* **Giải thích:**
  + **Mô hình thác nước (Waterfall Model)** có tính tuần tự, từng giai đoạn phải hoàn thành trước khi chuyển sang giai đoạn tiếp theo, ít linh hoạt khi yêu cầu thay đổi.
  + **Mô hình lặp và tăng trưởng (Incremental and Iterative Model)** phát triển phần mềm theo từng đợt nhỏ, mỗi đợt hoàn thiện thêm một phần tính năng, giúp thích nghi nhanh với thay đổi của khách hàng.

**Câu hỏi 10: Mô hình nào có khả năng thích nghi tốt nhất với sự thay đổi của yêu cầu khách hàng?**

* **Đáp án đúng: D. Mô hình tiến trình linh hoạt (Agile Model)**
* **Giải thích:**
  + **Mô hình Agile** được thiết kế để thích nghi với sự thay đổi nhanh chóng của yêu cầu khách hàng.
  + Các nhóm phát triển phần mềm làm việc theo **các vòng lặp ngắn (iterations)** và có thể điều chỉnh sản phẩm sau mỗi vòng để phù hợp với yêu cầu thực tế.
  + So với **mô hình thác nước** (cứng nhắc), **xoắn ốc** (quản lý rủi ro) hay **xây và sửa** (không có kế hoạch cụ thể), Agile mang lại sự linh hoạt cao nhất.

Part 2: Câu hỏi trả lời ngắn

**1. Pha lấy yêu cầu là gì và có vai trò gì trong vòng đời phát triển phần mềm?**

* **Trả lời:** Pha lấy yêu cầu là giai đoạn đầu tiên trong vòng đời phát triển phần mềm, nơi nhóm phát triển thu thập và phân tích các yêu cầu từ khách hàng hoặc người dùng.
* **Vai trò:**
  + Xác định chính xác mong muốn của khách hàng.
  + Làm cơ sở cho việc thiết kế và lập kế hoạch phát triển phần mềm.
* **Ví dụ:** Trong một dự án phát triển ứng dụng ngân hàng di động, nhóm phát triển sẽ thu thập yêu cầu như **chuyển tiền nhanh**, **thanh toán hóa đơn**, **đăng nhập bằng vân tay**, v.v.

**2. Mô hình thác nước hoạt động như thế nào?**

* **Trả lời:** Mô hình **thác nước (Waterfall)** là một phương pháp phát triển phần mềm tuyến tính, trong đó mỗi pha phải hoàn thành trước khi chuyển sang pha tiếp theo.
* **Các bước:**
  1. Lấy yêu cầu
  2. Thiết kế
  3. Lập trình
  4. Kiểm thử
  5. Triển khai
  6. Bảo trì
* **Ví dụ:** Một công ty phát triển phần mềm cho bệnh viện có thể sử dụng mô hình thác nước vì hệ thống cần sự **ổn định cao và ít thay đổi yêu cầu**.

**3. Mô hình lặp và tăng trưởng khác gì so với mô hình thác nước?**

* **Trả lời:**
  + **Mô hình thác nước** là tuyến tính, không thể quay lại giai đoạn trước.
  + **Mô hình lặp và tăng trưởng** phát triển phần mềm theo từng **đợt nhỏ**, có thể cập nhật và cải tiến dựa trên phản hồi người dùng.
* **Ví dụ:** Facebook liên tục cập nhật tính năng **Reels, Marketplace** mà không cần chờ hoàn thành toàn bộ hệ thống.

**4. Mục tiêu của pha bảo trì là gì?**

* **Trả lời:** Pha bảo trì nhằm **sửa lỗi, tối ưu hiệu suất, cập nhật tính năng mới** hoặc đảm bảo phần mềm hoạt động ổn định trong môi trường thay đổi.
* **Ví dụ:**
  + Google Chrome cập nhật **vá lỗi bảo mật** và **cải thiện tốc độ tải trang** trong mỗi phiên bản mới.

**5. Mô hình xây và sửa có nhược điểm gì?**

* **Trả lời:**
  + Thiếu kế hoạch rõ ràng, dễ phát sinh lỗi.
  + Chi phí bảo trì cao do không có quy trình kiểm thử chặt chẽ.
  + Phù hợp với dự án nhỏ nhưng khó mở rộng.
* **Ví dụ:** Một lập trình viên viết nhanh một ứng dụng chat nhưng **không kiểm thử kỹ**, khiến người dùng gặp lỗi gửi tin nhắn chậm hoặc không gửi được.

**6. Mô hình bản mẫu nhanh là gì?**

* **Trả lời:** Mô hình **bản mẫu nhanh (Prototyping Model)** tạo một phiên bản thử nghiệm nhanh chóng để lấy phản hồi từ khách hàng trước khi phát triển bản chính thức.
* **Ví dụ:** Khi thiết kế giao diện mới cho Shopee, nhóm UI/UX tạo bản **Prototype** để khách hàng đánh giá **trải nghiệm mua hàng** trước khi triển khai thực tế.

**7. Pha giải thể là gì?**

* **Trả lời:** Pha giải thể là giai đoạn **kết thúc vòng đời của phần mềm**, có thể bao gồm **gỡ bỏ hệ thống, chuyển dữ liệu sang hệ thống mới hoặc đóng dự án**.
* **Ví dụ:**
  + Microsoft đã **ngừng hỗ trợ Windows 7** vào năm 2020, yêu cầu người dùng nâng cấp lên Windows 10.

**8. Mô hình xoắn ốc là gì?**

* **Trả lời:** Mô hình **xoắn ốc (Spiral Model)** kết hợp mô hình thác nước và mô hình lặp, tập trung vào **quản lý rủi ro**. Mỗi vòng xoắn bao gồm:
  1. Phân tích rủi ro
  2. Thiết kế
  3. Lập trình
  4. Kiểm thử
* **Ví dụ:**
  1. Hệ thống **phần mềm quân sự** thường sử dụng mô hình này để đảm bảo tính **an toàn, bảo mật** và **giảm rủi ro** khi phát triển.

**9. Tại sao mô hình tiến trình linh hoạt được đánh giá cao?**

* **Trả lời:** Mô hình **tiến trình linh hoạt (Agile)** được đánh giá cao vì:
  + **Phù hợp với dự án thay đổi liên tục.**
  + **Tăng tốc độ phát triển** do làm việc theo vòng lặp ngắn.
  + **Tạo ra sản phẩm sớm** và có thể cải tiến liên tục.
* **Ví dụ:**
  + Công ty startup công nghệ như **Zalo, Grab** thường áp dụng Agile để nhanh chóng thử nghiệm và tối ưu ứng dụng.

**10. Điểm khác biệt chính giữa mô hình mã nguồn mở và các mô hình khác là gì?**

* **Trả lời:** Mô hình mã nguồn mở khác với các mô hình khác ở các điểm:
  + **Cộng đồng phát triển** thay vì một nhóm nội bộ.
  + **Mã nguồn công khai**, ai cũng có thể đóng góp và sử dụng.
  + **Cập nhật liên tục**, không phụ thuộc vào một công ty duy nhất.
* **Ví dụ:**
  + **Linux, Android, MySQL** là những phần mềm mã nguồn mở, cho phép lập trình viên trên toàn thế giới đóng góp cải tiến.

Part 3: Câu hỏi thảo luận nhóm:

**3. Tại sao mô hình xây và sửa không phù hợp với các dự án lớn?**

* **Trả lời:**
  + Mô hình **xây và sửa (Build and Fix)** không có kế hoạch rõ ràng, không có quy trình phát triển chặt chẽ.
  + Dễ dẫn đến **mã nguồn rối rắm**, khó mở rộng và bảo trì.
  + Không có giai đoạn kiểm thử bài bản, gây ra nhiều **lỗi nghiêm trọng khi hệ thống phát triển lớn**.
  + Chi phí sửa lỗi và bảo trì rất cao vì không có thiết kế ban đầu tốt.
* **Ví dụ:**
  + Một startup nhỏ phát triển ứng dụng **quản lý nhân sự nội bộ** bằng cách code nhanh mà không có kiến trúc rõ ràng. Khi công ty mở rộng, hệ thống không thể đáp ứng số lượng nhân viên lớn, gây lỗi liên tục và khó nâng cấp.
  + Ngược lại, **Facebook hoặc Google** không thể áp dụng mô hình này vì hệ thống của họ quá lớn, cần một kiến trúc **chặt chẽ ngay từ đầu**.

4. So sánh giữa mô hình bản mẫu nhanh và mô hình tiến trình linh hoạt

|  |  |  |
| --- | --- | --- |
| Tiêu chí | Mô hình bản mẫu nhanh (Prototyping Model) | Mô hình tiến trình linh hoạt (Agile Model) |
| Mục đích | Xây dựng bản thử nghiệm nhanh để thu thập phản hồi từ khách hàng. | Phát triển phần mềm theo vòng lặp nhỏ, có thể thay đổi nhanh chóng. |
| Cách thực hiện | Tạo ra một bản mẫu đơn giản để khách hàng đánh giá. | Chia nhỏ phần mềm thành nhiều phần, mỗi phần có thể triển khai độc lập. |
| Tính linh hoạt | Tập trung vào thay đổi giao diện và chức năng dựa trên phản hồi. | Rất linh hoạt, có thể thay đổi yêu cầu bất cứ lúc nào. |
| Thích hợp với loại dự án | Các dự án cần kiểm tra tính khả thi nhanh như UI/UX, ứng dụng web đơn giản. | Các dự án phần mềm phức tạp, cần phát triển lâu dài và có thể thay đổi thường xuyên. |
| Ví dụ | Thiết kế một giao diện mới cho ứng dụng mua sắm Shopee, khách hàng có thể dùng thử trước khi triển khai chính thức. | Phát triển ứng dụng Grab, mỗi tháng có thể thêm tính năng như thanh toán ví điện tử, đặt xe liên tỉnh mà không ảnh hưởng đến hệ thống cũ. |

Part 4: Câu hỏi tình huống

**3. Trong quá trình phát triển phần mềm theo mô hình tiến trình linh hoạt, khách hàng không đưa ra phản hồi kịp thời. Đội phát triển nên xử lý ra sao?**

Trong mô hình **tiến trình linh hoạt (Agile)**, phản hồi của khách hàng rất quan trọng để điều chỉnh sản phẩm theo nhu cầu thực tế.

Nếu khách hàng không phản hồi kịp thời, đội phát triển có thể:

* + 1. **Sử dụng Product Owner** để thay mặt khách hàng ra quyết định tạm thời.
    2. **Ghi nhận lại các quyết định quan trọng**, giúp khách hàng hiểu những thay đổi đã thực hiện.
    3. **Tiến hành thử nghiệm nội bộ** để đảm bảo chất lượng sản phẩm mà không cần chờ phản hồi ngay lập tức.
    4. **Chia nhỏ các tính năng** để tiếp tục phát triển phần mềm mà không ảnh hưởng đến tiến độ.

**Ví dụ:** Một nhóm phát triển phần mềm cho một **ứng dụng thương mại điện tử**. Nếu khách hàng không phản hồi về giao diện giỏ hàng, nhóm có thể **tham khảo xu hướng UI/UX từ các ứng dụng phổ biến** như Shopee, Lazada để tự điều chỉnh trước khi nhận phản hồi chính thức.

**4. Khách hàng yêu cầu bổ sung một số tính năng mới khi phần mềm đã bước vào pha cài đặt. Nên áp dụng mô hình nào để xử lý tốt nhất yêu cầu này?**

Trong trường hợp này, **mô hình tiến trình linh hoạt (Agile) hoặc mô hình xoắn ốc (Spiral Model)** sẽ phù hợp hơn vì:

1. **Agile:** Cho phép thay đổi liên tục ngay cả khi sản phẩm đã gần hoàn thành
2. **Xoắn ốc:** Hỗ trợ đánh giá rủi ro trước khi thêm tính năng mới.

Các mô hình như **thác nước (Waterfall)** không phù hợp vì **không cho phép thay đổi khi đã đến giai đoạn cài đặt**.

**Ví dụ:**

1. Công ty phát triển phần mềm kế toán đã hoàn tất phần mềm và bắt đầu triển khai cho khách hàng. **Khách hàng yêu cầu thêm tính năng tạo báo cáo thuế tự động**.
2. Nếu dùng **mô hình thác nước**, nhóm phát triển phải đợi đến phiên bản sau mới có thể cập nhật.
3. Nếu dùng **Agile**, họ có thể phát triển nhanh bản cập nhật và thêm vào ngay khi triển khai.