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

**THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN 2**





**BÁO CÁO**

**MÔN HỌC: NHẬP MÔN CÔNG NGHỆ PHẦN MỀM**

**BÀI TẬP CHƯƠNG 2: TIẾN TRÌNH PHẦN MỀM**

|  |  |
| --- | --- |
| **Giáo viên hướng dẫn:** | **Gv. Châu Văn Vân** |
| **Nhóm sinh viên thực hiện:** | **5** |
| **Lớp:** | **D22CQAT01** |
| **Thành viên nhóm:** | **Phạm Đình Nam - N21DCCN054**  **Vũ Huy Hùng – N20DCCN020** |
|  | **Hồ Xuân Thiện – N22DCAT051**  **Lê Anh Dũng – N22DCAT011**  **Nguyễn Duyên Duy – N22DCAT013** |

***TP.HCM, 2/2025***

LỜI CẢM ƠN

Em chân thành cảm ơn Thầy Châu Văn Vân, người đã tận tình hướng dẫn và giúp đỡ em trong quá trình học môn Nhập Môn Công Nghệ Phần Mềm. Kiến thức và kỹ năng thu được từ môn học này cũng sẽ hỗ trợ em trong việc làm việc nhóm và quản lý dự án, xây dựng phần mềm trong tương lai.

Em cũng muốn bày tỏ lòng biết ơn đến Ban lãnh đạo và các giảng viên tại Học Viện Công Nghệ Bưu Chính Viễn Thông, đã tạo điều kiện và cung cấp cơ sở vật chất để chúng em có môi trường học tập và rèn luyện.

Dù em nhận thức rằng đề tài của mình có những hạn chế và thiếu sót do kiến thức và kinh nghiệm còn hạn chế, và thời gian nghiên cứu có hạn, nhưng em hy vọng nhận được sự đóng góp ý kiến của thầy để có thêm kiến thức và kinh nghiệm, từ đó hoàn thiện đề tài một cách tốt nhất.

Em xin trân trọng cảm ơn!

*Thành phố Hồ Chí Minh, tháng 2 năm 2025*

# BẢNG PHÂN CÔNG CÔNG VIỆC

SINH VIÊN: Nguyễn Duyên Duy

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ tên** | **Phụ trách công việc** | **Thời gian** |
| 1 | Phạm Đình Nam  (Nhóm trưởng) | Phân công nhiệm vụ.  Thực hiện:   * Câu 1.1, 1.2 * Câu 2.1, 2.2 * Câu 3 * Câu 4.1, 4.2 | Phân công nhiệm vụ:  Ngày 24/2/2025.  Thực hiện:  Từ ngày 24/2.  Đến ngày 26/2.  Tổng hợp câu 3:  Ngày 27/2/2025  \*Note: Câu 3 là câu thảo luận nhóm nên cần tổng hợp lại. |
| 2 | Vũ Huy Hùng | Thực hiện:   * Câu 1.9, 1.10 * Câu 2.9, 2.10 * Câu 3 * Câu 4.9, 4.10 |
| 3 | Hồ Xuân Thiện | Thực hiện:   * Câu 1.7, 1.8 * Câu 2.7, 2.8 * Câu 3 * Câu 4.7, 4.8 |
| 4 | Lê Anh Dũng | Thực hiện:   * Câu 1.5, 1.6 * Câu 2.5, 2.6 * Câu 3 * Câu 4.5, 4.6 |
| 5 | Nguyễn Duyên Duy | Thực hiện:   * Câu 1.3, 1.4 * Câu 2.3, 2.4 * Câu 3 * Câu 4.3, 4.4 |

**MỤC LỤC**

[BẢNG PHÂN CÔNG CÔNG VIỆC 3](#_heading=h.1fob9te)

[PHẦN 1: CÂU HỎI TRẮC NGHIỆM 5](#_heading=h.3znysh7)

[Câu 1.1:Câu hỏi 1: Workflow nào trong tiến trình phát triển phần mềm chịu trách nhiệm thu thập yêu cầu từ khách hàng? 6](#_heading=h.2et92p0)

[Câu 1.2:Pha nào trong tiến trình thống nhất (Unified Process) tập trung vào việc phân tích rủi ro và xây dựng kiến trúc ban đầu? 6](#_heading=h.tyjcwt)

[Câu 1.3: Mô hình CMM mức nào yêu cầu quy trình phát triển phần mềm phải được quản lý định lượng? 6](#_heading=h.3dy6vkm)

[Câu 1.4: Các pha trong tiến trình thống nhất bao gồm: 6](#_heading=h.1t3h5sf)

[Câu 1.5: Trong tiến trình thống nhất, workflow nào thực hiện sau cùng? 6](#_heading=h.4d34og8)

[Câu 1.6:Mô hình CMM mức 1 có đặc điểm gì? 6](#_heading=h.2s8eyo1)

[Câu 1.7:Tiến trình thống nhất là một ví dụ của mô hình nào? 7](#_heading=h.17dp8vu)

[Câu 1.8:Trong mô hình CMM mức 5, quy trình phát triển phần mềm có đặc điểm gì? 7](#_heading=h.3rdcrjn)

[Câu 1.9: Workflow thiết kế bao gồm việc thực hiện hoạt động nào? 7](#_heading=h.108q7rt43sl4)

[Câu 1.10: CMM viết tắt của cụm từ nào? 7](#_heading=h.cjuxglpmk38)

[PHẦN 2: CÂU TRẢ LỜI NGẮN 7](#_heading=h.35nkun2)

[Câu 2.1: Pha khởi đầu trong tiến trình thống nhất là gì? 7](#_heading=h.1ksv4uv)

[Câu 2.2: Mục tiêu của workflow lấy yêu cầu là gì? 8](#_heading=h.44sinio)

[Câu 2.3: Tiến trình thống nhất gồm bao nhiêu pha chính? 8](#_heading=h.2jxsxqh)

[Câu 2.4: Sự khác nhau giữa CMM mức 2 và mức 3 là gì? 9](#_heading=h.z337ya)

[Câu 2.6:Mô hình CMM có bao nhiêu mức? 10](#_heading=h.1y810tw)

[Câu 2.7:Khác biệt giữa mô hình thác nước và mô hình lặp là gì? 11](#_heading=h.4i7ojhp)

[Câu 2.8:Tiến trình thống nhất có phải là mô hình lặp không? 12](#_heading=h.2xcytpi)

[Câu 2.9: Mục đích của workflow thiết kế là gì? 12](#_heading=h.sm6w4jn5tgv9)

[Câu 2.10: CMM mức 5 tập trung vào điều gì? 13](#_heading=h.h86tn88idm4f)

[PHẦN 3: CÂU HỎI THẢO LUẬN NHÓM 13](#_heading=h.2bn6wsx)

[Câu 3.1: Thảo luận về vai trò của từng workflow trong tiến trình phát triển phần mềm. 13](#_heading=h.qsh70q)

[Câu 3.2: Phân biệt mô hình vòng đời thác nước và tiến trình thống nhất. 13](#_heading=h.3as4poj)

[Câu 3.3: Thảo luận về các ưu và nhược điểm của mô hình lặp và tăng trưởng. 14](#_heading=h.1pxezwc)

[Câu 3.4:Vì sao mô hình CMM được sử dụng rộng rãi trong quản lý chất lượng phần mềm? 15](#_heading=h.49x2ik5)

[Câu 3.5:Thảo luận về các khó khăn khi áp dụng mô hình CMM trong thực tế. 16](#_heading=h.2p2csry)

[Câu 3.6:Giải pháp để cải tiến quy trình phát triển phần mềm. 17](#_heading=h.147n2zr)

[Câu 3.7:Phân tích ưu điểm của việc áp dụng tiến trình thống nhất trong các dự án lớn. 17](#_heading=h.3o7alnk)

[Câu 3.8:Thảo luận về sự cần thiết của việc kiểm thử trong từng pha của tiến trình thống nhất. 19](#_heading=h.23ckvvd)

[Câu 3.9: So sánh giữa mô hình CMM mức 4 và mức 5. 20](#_heading=h.ihv636)

[Câu 3.10: Đề xuất cách tổ chức hoạt động nhóm trong workflow lấy yêu cầu 21](#_heading=h.32hioqz)

[PHẦN 4: CÂU HỎI TÌNH HUỐNG 21](#_heading=h.1hmsyys)

[Câu 4.1: Một công ty phát triển phần mềm gặp khó khăn khi yêu cầu của khách hàng liên tục thay đổi trong pha xây dựng. Đội phát triển nên làm gì để giải quyết vấn đề này? 21](#_heading=h.41mghml)

[Câu 4.2: Trong pha chuyển giao của tiến trình thống nhất, khách hàng yêu cầu bổ sung thêm tính năng mới. Đội phát triển nên xử lý ra](#_heading=h.2grqrue) [sao? 21](#_heading=h.5zxix9814z7m)

[Câu 4.3:Dự án phát triển phần mềm bị trễ tiến độ do lỗi phát sinh liên tục trong quá trình kiểm thử. Là trưởng dự án, bạn sẽ làm gì? 22](#_heading=h.vx1227)

[Câu 4.4:Trong workflow thiết kế, kiến trúc sư phần mềm muốn thay đổi thiết kế ban đầu để cải thiện hiệu suất. Đội phát triển nên xử lý thế nào? 23](#_heading=h.zi14v8g91uzc)

[Câu 4.5:Khách hàng yêu cầu rút ngắn thời gian phát triển dự án mà không thay đổi yêu cầu. Đội phát triển nên phản ứng ra sao? 25](#_heading=h.1v1yuxt)

[Câu 4.6:Một công ty nhỏ muốn áp dụng mô hình CMM nhưng gặp khó khăn do thiếu nguồn lực. Hãy đề xuất giải pháp. 26](#_heading=h.4f1mdlm)

[Câu 4.7:Trong workflow lấy yêu cầu, khách hàng cung cấp thông tin không rõ ràng. Đội phát triển cần làm gì? 26](#_heading=h.2u6wntf)

[Câu 4.8: Một dự án gặp rủi ro cao trong pha khởi đầu do thiếu tài liệu yêu cầu rõ ràng. Đội phát triển nên làm gì? 28](#_heading=h.19c6y18)

[Câu 4.9: Dự án phần mềm lớn có nhiều nhóm phát triển ở các địa điểm khác nhau. Làm thế nào để đảm bảo các nhóm phối hợp hiệu quả? 30](#_heading=h.3tbugp1)

[Câu 4.10: Một công ty phát triển phần mềm gặp khó khăn trong việc quản lý quy trình do không có chuẩn hóa. Hãy đề xuất giải pháp. 30](#_heading=h.28h4qwu)

# PHẦN 1: CÂU HỎI TRẮC NGHIỆM

## Câu 1.1:Câu hỏi 1: Workflow nào trong tiến trình phát triển phần mềm chịu trách nhiệm thu thập yêu cầu từ khách hàng?

A. Workflow thiết kế

B. Workflow lấy yêu cầu

C. Workflow kiểm thử

D. Workflow triển khai

Đáp án: B

## Câu 1.2:Pha nào trong tiến trình thống nhất (Unified Process) tập trung vào việc phân tích rủi ro và xây dựng kiến trúc ban đầu?

A. Pha khởi đầu

B. Pha làm rõ

C. Pha xây dựng

D. Pha chuyển giao

Đáp án: B

## Câu 1.3: Mô hình CMM mức nào yêu cầu quy trình phát triển phần mềm phải được quản lý định lượng?

A. Mức 2

B. Mức 3

C. Mức 4

D. Mức 5

Đáp án: C

## Câu 1.4: Các pha trong tiến trình thống nhất bao gồm:

A. Lấy yêu cầu, phân tích, thiết kế, kiểm thử .

B. Khởi đầu, làm rõ, xây dựng, chuyển giao .

C. Lập kế hoạch, thiết kế, phát triển, bảo trì .

D. Phân tích, kiểm thử, triển khai, bảo trì.

Đáp án: B

## Câu 1.5: Trong tiến trình thống nhất, workflow nào thực hiện sau cùng?

A. Workflow phân tích

B. Workflow thiết kế

C. Workflow cài đặt

D. Workflow kiểm thử

Đáp án: D

## Câu 1.6:Mô hình CMM mức 1 có đặc điểm gì?

A. Quy trình được định nghĩa rõ ràng

B. Quy trình được kiểm soát và đo lường

**C. Quy trình không ổn định, phụ thuộc vào cá nhân**

D. Quy trình liên tục được tối ưu hóa

Đáp án: C

## Câu 1.7:Tiến trình thống nhất là một ví dụ của mô hình nào?

**A. Mô hình vòng đời thác nước**

B. Mô hình lặp và tăng trưởng

C. Mô hình mã nguồn mở

D. Mô hình Agile

Đáp án: A

## 

## Câu 1.8:Trong mô hình CMM mức 5, quy trình phát triển phần mềm có đặc điểm gì?

**A. Quy trình được cải tiến liên tục**

B. Quy trình chỉ định nghĩa cơ bản

C. Quy trình chưa được quản lý

D. Quy trình chỉ tập trung vào bảo trì

Đáp án: A

## 

## Câu 1.9: Workflow thiết kế bao gồm việc thực hiện hoạt động nào?

A. Thu thập yêu cầu

B. Lập kế hoạch dự án

**C. Thiết kế kiến trúc và chi tiết hệ thống**

D. Kiểm thử tích hợp

Đáp án: C

## Câu 1.10: CMM viết tắt của cụm từ nào?

A. Configuration Management Model

**B. Capability Maturity Model**

C. Continuous Maintenance Model

D. Complex Management Model

Đáp án: B

# PHẦN 2: CÂU TRẢ LỜI NGẮN

## Câu 2.1: Pha khởi đầu trong tiến trình thống nhất là gì?

Pha khởi đầu (Inception Phase) trong tiến trình thống nhất (Unified Process) là giai đoạn đầu tiên của vòng đời phát triển phần mềm. Đây là giai đoạn quan trọng đặt nền móng cho toàn bộ dự án. Trong pha này, các hoạt động chính bao gồm:

1. Xác định phạm vi dự án: Xác định ranh giới và mục tiêu tổng thể của dự án.
2. Xác định tầm nhìn sản phẩm: Phát triển tầm nhìn chung về những gì cần đạt được.
3. Xác định các bên liên quan (stakeholders): Xác định và phân tích nhu cầu của tất cả các bên liên quan.
4. Đánh giá tính khả thi: Phân tích tính khả thi về kinh tế, kỹ thuật và tổ chức của dự án.
5. Xác định các trường hợp sử dụng (use cases) cốt lõi: Xác định các tính năng quan trọng nhất của hệ thống.
6. Ước tính nguồn lực: Đưa ra ước tính ban đầu về chi phí, thời gian và nguồn lực cần thiết.
7. Xác định rủi ro ban đầu: Phát hiện các rủi ro chính có thể ảnh hưởng đến dự án.
8. Xây dựng kế hoạch dự án sơ bộ: Thiết lập lịch trình và các cột mốc ban đầu.

Mục tiêu cuối cùng của pha khởi đầu là đưa ra quyết định có tiếp tục dự án hay không dựa trên các thông tin đã thu thập. Pha này thường ngắn hơn nhiều so với các pha tiếp theo và kết thúc với cột mốc "Lifecycle Objectives" (Mục tiêu vòng đời), nơi các bên liên quan đánh giá tính khả thi và đồng ý về phạm vi và tầm nhìn của dự án.

## Câu 2.2: Mục tiêu của workflow lấy yêu cầu là gì?

Mục tiêu của workflow lấy yêu cầu (Requirements Workflow) trong tiến trình thống nhất (Unified Process) bao gồm:

1. Thiết lập và duy trì sự đồng thuận giữa người dùng và nhóm phát triển về những gì hệ thống cần làm
2. Xác định và định nghĩa rõ ràng các yêu cầu chức năng và phi chức năng của hệ thống
3. Cung cấp cơ sở cho việc lập kế hoạch nội dung kỹ thuật của các bản phát hành
4. Cung cấp cơ sở cho việc ước tính thời gian và nguồn lực cần thiết để phát triển hệ thống
5. Định nghĩa giao diện người dùng của hệ thống, tập trung vào nhu cầu và mục tiêu của người dùng
6. Phát triển tầm nhìn rõ ràng về phạm vi và giới hạn của hệ thống
7. Tạo ra các mô hình trường hợp sử dụng (use cases) để mô tả hành vi của hệ thống từ góc độ người dùng
8. Tạo cơ sở để xác định và quản lý những thay đổi về yêu cầu trong quá trình phát triển
9. Hỗ trợ cho việc kiểm thử bằng cách cung cấp các tiêu chí để đánh giá hiệu suất và chức năng của hệ thống

Workflow lấy yêu cầu được thực hiện xuyên suốt toàn bộ vòng đời phát triển phần mềm, nhưng đặc biệt tập trung vào pha khởi đầu (Inception) và pha làm rõ (Elaboration) của tiến trình thống nhất.

## Câu 2.3: Tiến trình thống nhất gồm bao nhiêu pha chính?

* **Khởi đầu (Inception):**Xác định mục tiêu, phạm vi dự án và đánh giá khả năng khả thi. Giai đoạn này giúp thiết lập cơ sở cho toàn bộ dự án.
* **Làm rõ (Elaboration):**Phân tích chi tiết các yêu cầu, xây dựng kiến trúc hệ thống và xác định các rủi ro chính. Mục tiêu là làm rõ và củng cố các nền tảng của dự án.
* **Xây dựng (Construction):**Phát triển và lập trình phần mềm, tích hợp các thành phần và thực hiện kiểm thử nội bộ. Giai đoạn này tập trung vào việc tạo ra sản phẩm phần mềm theo kế hoạch.
* **Chuyển giao (Transition):**Triển khai phần mềm vào môi trường thực tế, thực hiện kiểm thử trên môi trường khách hàng và bàn giao sản phẩm cuối cùng. Đây là giai đoạn để đảm bảo sản phẩm hoạt động ổn định và được người dùng chấp nhận.

## Câu 2.4: Sự khác nhau giữa CMM mức 2 và mức 3 là gì?

## 

**Phạm vi quy trình:**

* *Mức 2:* Quy trình được quản lý ở cấp độ dự án, có thể khác nhau giữa các dự án.
* *Mức 3:* Quy trình được chuẩn hóa và áp dụng đồng nhất trên toàn tổ chức.

**Quản lý và kiểm soát:**

* *Mức 2:* Tập trung vào quản lý dự án riêng lẻ, với các quy trình được thiết lập cho từng dự án cụ thể.
* *Mức 3:* Tập trung vào việc thiết lập và duy trì các quy trình chuẩn hóa trên toàn tổ chức, đảm bảo tính nhất quán và hiệu quả trong việc thực hiện các dự án.

**Câu 2.5:Workflow kiểm thử có nhiệm vụ gì?**

**Các nhiệm vụ chính của Workflow kiểm thử:**

1. **Xác minh và xác thực phần mềm**

**Xác minh (Verification):** Kiểm tra xem phần mềm có được phát triển đúng theo thiết kế và yêu cầu kỹ thuật không.

**Xác thực (Validation):** Đảm bảo phần mềm đáp ứng đúng nhu cầu của khách hàng và người dùng.

1. **Phát hiện và sửa lỗi (Bug Detection & Debugging)**

Tìm ra lỗi trong phần mềm và báo cáo cho nhóm phát triển.

Kiểm tra lại sau khi lỗi được sửa để đảm bảo chúng không tái diễn.

1. **Đánh giá chất lượng phần mềm**

Kiểm tra các tiêu chí như **hiệu suất, bảo mật, tính ổn định, khả năng mở rộng**.

Đảm bảo phần mềm chạy ổn định trên nhiều môi trường khác nhau.

1. **Thực hiện kiểm thử các loại khác nhau**

**Kiểm thử đơn vị (Unit Testing):** Kiểm tra từng module riêng lẻ.

**Kiểm thử tích hợp (Integration Testing):** Đánh giá sự tương tác giữa các module.

**Kiểm thử hệ thống (System Testing):** Kiểm tra toàn bộ hệ thống.

**Kiểm thử chấp nhận (Acceptance Testing):** Đánh giá xem phần mềm có đáp ứng yêu cầu khách hàng không.

1. **Tự động hóa và báo cáo kiểm thử**

Sử dụng công cụ kiểm thử tự động (Selenium, JUnit, TestNG...) để tăng hiệu quả.

Ghi lại kết quả kiểm thử và tạo báo cáo để nhóm phát triển khắc phục lỗi.

## Câu 2.6:Mô hình CMM có bao nhiêu mức?

**Mô hình CMM có 5 mức độ**

**Mức 1 – Initial (Khởi đầu)**

Quy trình không ổn định, phụ thuộc vào cá nhân.

Thiếu sự kiểm soát và lập kế hoạch rõ ràng.

Dự án dễ bị chậm tiến độ, vượt ngân sách hoặc chất lượng kém.

**Mức 2 – Managed (Có thể quản lý)**

Quy trình được định nghĩa sơ bộ, có kế hoạch và quản lý cơ bản.

Quản lý dự án dựa trên kinh nghiệm trước đó.

Bắt đầu kiểm soát chi phí, thời gian và chất lượng.

**Mức 3 – Defined (Được định nghĩa rõ ràng)**

Quy trình được chuẩn hóa, có tài liệu và hướng dẫn cụ thể.

Mọi dự án đều tuân theo quy trình chuẩn của tổ chức.

Dễ dàng tái sử dụng quy trình cho các dự án khác.

**Mức 4 – Managed (Được quản lý chặt chẽ)**

Quy trình được đo lường và kiểm soát bằng các chỉ số cụ thể.

Sử dụng dữ liệu thống kê để dự đoán hiệu suất dự án.

Cải thiện chất lượng phần mềm bằng cách kiểm soát rủi ro tốt hơn.

**Mức 5 – Optimizing (Tối ưu hóa liên tục)**

Liên tục cải tiến quy trình để tăng hiệu suất và chất lượng.

Ứng dụng các công nghệ và phương pháp mới để nâng cao hiệu quả.

Có khả năng thích ứng linh hoạt với thay đổi trong yêu cầu và môi trường phát triển.

## Câu 2.7:Khác biệt giữa mô hình thác nước và mô hình lặp là gì?

|  |  |  |
| --- | --- | --- |
| **Tiêu chí** | **Mô hình thác nước (Waterfall Model)** | **Mô hình lặp (Iterative Model)** |
| **Cách tiếp cận** | Tuần tự, từng bước, không quay lại giai đoạn trước. | Lặp lại nhiều lần, cải tiến dần dần qua mỗi vòng lặp. |
| **Quy trình phát triển** | Bao gồm các giai đoạn cố định: **Yêu cầu → Thiết kế → Triển khai → Kiểm thử → Bảo trì**. | Chia nhỏ thành nhiều vòng lặp, mỗi vòng gồm các bước: **Phân tích → Thiết kế → Xây dựng → Kiểm thử**. |
| **Tính linh hoạt** | Cứng nhắc, khó thay đổi khi đã sang giai đoạn tiếp theo. | Linh hoạt, có thể điều chỉnh yêu cầu và tính năng trong từng vòng lặp. |
| **Thời gian phản hồi** | Phải hoàn thành toàn bộ trước khi có sản phẩm. | Có thể cung cấp sản phẩm sớm hơn với các phiên bản nhỏ. |
| **Rủi ro** | Cao, vì lỗi chỉ phát hiện ở cuối quá trình. | Thấp hơn, vì có thể phát hiện lỗi sớm và sửa chữa trong các lần lặp sau. |
| **Chi phí sửa lỗi** | Cao, do lỗi phát hiện muộn. | Thấp hơn, vì lỗi có thể được sửa trong từng vòng lặp. |
| **Ứng dụng phù hợp** | Dự án nhỏ, yêu cầu rõ ràng, ít thay đổi. | Dự án lớn, yêu cầu có thể thay đổi, cần phản hồi nhanh từ người dùng. |

## Câu 2.8:Tiến trình thống nhất có phải là mô hình lặp không?

Có, Tiến trình thống nhất (UP - Unified Process) là một mô hình lặp và tăng trưởng (Iterative and Incremental Model).

**Giải thích:**

**Tiến trình thống nhất** không phát triển phần mềm theo cách tuyến tính như **mô hình thác nước**, mà thay vào đó, phần mềm được phát triển qua nhiều **vòng lặp**. Mỗi vòng lặp cung cấp một phần hoàn thiện hơn của hệ thống, giúp phát hiện lỗi và cải tiến liên tục.

**Đặc điểm của Tiến trình thống nhất (UP):**

1. **Lặp đi lặp lại**:
   * Dự án được chia thành nhiều vòng lặp nhỏ, mỗi vòng tạo ra một phiên bản phần mềm tốt hơn.
2. **Tăng trưởng dần dần**:
   * Phần mềm không hoàn thành ngay từ đầu mà được cải tiến dần qua các lần lặp.
3. **Gồm 4 giai đoạn chính**:
   * **Khởi đầu (Inception)** → Xác định yêu cầu ban đầu.
   * **Lập kế hoạch (Elaboration)** → Thiết kế kiến trúc phần mềm.
   * **Xây dựng (Construction)** → Phát triển, lập trình và kiểm thử.
   * **Chuyển giao (Transition)** → Hoàn thiện, triển khai và bảo trì.

**Vì sao UP thuộc mô hình lặp?**

✅ **Có tính lặp**: Phần mềm được phát triển qua từng chu kỳ nhỏ, không phải hoàn thiện ngay từ đầu.  
✅ **Có tính tăng trưởng**: Mỗi vòng lặp làm cho phần mềm hoàn thiện hơn.  
✅ **Cho phép thay đổi yêu cầu**: Nếu yêu cầu thay đổi, có thể điều chỉnh trong vòng lặp tiếp theo.

📌 **Kết luận**: **Tiến trình thống nhất (UP) là một dạng của mô hình lặp và tăng trưởng**, không phải mô hình tuyến tính như thác nước. ✅

## Câu 2.9: Mục đích của workflow thiết kế là gì?

Mục đích của workflow thiết kế là thiết kế chi tiết các thành phần phần mềm dựa trên kết quả phân tích.

* Xác định kiến trúc hệ thống và mối quan hệ giữa các thành phần.
* Chuyển đổi các yêu cầu đã phân tích thành thiết kế có thể triển khai.
* Đảm bảo hệ thống có cấu trúc rõ ràng, dễ mở rộng và bảo trì.

## Câu 2.10: CMM mức 5 tập trung vào điều gì?

CMM mức 5 – Optimizing (Tối ưu hóa)

* Quy trình liên tục được cải tiến dựa trên phản hồi và dữ liệu.
* Mục tiêu là đạt được sự hoàn hảo trong phát triển phần mềm.

# PHẦN 3: CÂU HỎI THẢO LUẬN NHÓM

## Câu 3.1: Thảo luận về vai trò của từng workflow trong tiến trình phát triển phần mềm.

1. Workflow Lấy yêu cầu: Xác định nhu cầu người dùng, tạo ra các use case và đặc tả yêu cầu chức năng/phi chức năng.
2. Workflow Phân tích: Chuyển đổi yêu cầu thành mô hình phân tích chi tiết, xác định các lớp và mối quan hệ.
3. Workflow Thiết kế: Xây dựng kiến trúc hệ thống, thiết kế chi tiết các thành phần và cơ sở dữ liệu.
4. Workflow Triển khai (Implementation): Lập trình, chuyển thiết kế thành mã nguồn và tích hợp các thành phần.
5. Workflow Kiểm thử: Đảm bảo chất lượng thông qua các loại test, phát hiện và sửa lỗi.
6. Workflow Triển khai (Deployment): Đưa sản phẩm đến người dùng, bao gồm cài đặt, cấu hình và đào tạo.
7. Workflow Quản lý cấu hình và thay đổi: Kiểm soát phiên bản và quản lý thay đổi trong quá trình phát triển.
8. Workflow Quản lý dự án: Điều phối hoạt động, lập kế hoạch, quản lý nguồn lực và rủi ro.
9. Workflow Môi trường: Cung cấp công cụ và hỗ trợ kỹ thuật cho quá trình phát triển.

Các workflow này hoạt động song song và lặp lại qua các pha khác nhau (Khởi đầu, Làm rõ, Xây dựng, Chuyển giao), với mức độ tập trung khác nhau tùy thuộc vào từng giai đoạn của dự án.

## Câu 3.2: Phân biệt mô hình vòng đời thác nước và tiến trình thống nhất.

**Mô hình thác nước (Waterfall)**

* Đặc điểm cơ bản: Tuần tự, tuyến tính, mỗi giai đoạn hoàn thành trước khi bắt đầu giai đoạn tiếp theo
* Các pha: Yêu cầu → Thiết kế → Triển khai → Kiểm thử → Bảo trì
* Tính linh hoạt: Thấp, khó thay đổi sau khi đã hoàn thành một giai đoạn
* Tài liệu: Tạo tài liệu đầy đủ ở mỗi giai đoạn trước khi chuyển sang giai đoạn tiếp theo
* Tham gia của khách hàng: Chủ yếu ở giai đoạn đầu khi xác định yêu cầu và cuối cùng khi nghiệm thu
* Khả năng xử lý rủi ro: Hạn chế, rủi ro thường được phát hiện muộn trong quá trình
* Phù hợp với: Dự án nhỏ, yêu cầu ổn định, phạm vi rõ ràng

**Tiến trình thống nhất (Unified Process)**

* Đặc điểm cơ bản: Lặp lại, gia tăng, hướng use case và kiến trúc, tập trung vào rủi ro
* Các pha: Khởi đầu (Inception) → Làm rõ (Elaboration) → Xây dựng (Construction) → Chuyển giao (Transition)
* Tính linh hoạt: Cao, cho phép thay đổi và điều chỉnh liên tục
* Tài liệu: Phát triển dần, tập trung vào các tài liệu cần thiết cho từng vòng lặp
* Tham gia của khách hàng: Liên tục trong suốt dự án thông qua các vòng lặp phản hồi
* Khả năng xử lý rủi ro: Tốt, xác định và giải quyết rủi ro sớm, đặc biệt trong pha Làm rõ
* Workflow: Nhiều workflow song song (Yêu cầu, Phân tích, Thiết kế, Triển khai, Kiểm thử...) xuyên suốt các pha
* Phù hợp với: Dự án lớn, phức tạp, yêu cầu thay đổi, cần kiểm soát rủi ro

**Sự khác biệt chính**

1. Cấu trúc quy trình: Thác nước là tuyến tính, tuần tự; UP là lặp lại, gia tăng
2. Quản lý thay đổi: Thác nước khó thích ứng với thay đổi; UP chấp nhận và quản lý thay đổi
3. Sản phẩm ban đầu: Thác nước tạo ra sản phẩm cuối cùng sau khi hoàn thành toàn bộ quy trình; UP tạo ra phiên bản hoạt động sau mỗi vòng lặp
4. Kiểm thử: Thác nước kiểm thử ở cuối quy trình; UP tích hợp kiểm thử xuyên suốt
5. Tập trung rủi ro: Thác nước ít chú trọng; UP đặt quản lý rủi ro là trung tâm, đặc biệt trong pha Làm rõ

Tiến trình thống nhất khắc phục nhiều nhược điểm của mô hình thác nước bằng cách tích hợp các nguyên tắc của phát triển phần mềm linh hoạt, đồng thời vẫn duy trì cấu trúc và tài liệu cho các dự án lớn.

## Câu 3.3: Thảo luận về các ưu và nhược điểm của mô hình lặp và tăng trưởng.

**Ưu điểm:**

* **Phát hiện lỗi sớm:**Mỗi vòng lặp cho phép kiểm thử và sửa lỗi sớm, giúp giảm thiểu nguy cơ phát hiện lỗi ở giai đoạn sau, từ đó tiết kiệm thời gian và chi phí sửa chữa.
* **Linh hoạt và thích ứng:**Do được chia thành nhiều vòng lặp nhỏ, mô hình này cho phép điều chỉnh yêu cầu và thiết kế dựa trên phản hồi từ người dùng và thay đổi của thị trường.
* **Quản lý rủi ro hiệu quả:**Các vòng lặp giúp liên tục đánh giá và giảm thiểu rủi ro trong quá trình phát triển, vì mỗi vòng đều có cơ chế kiểm tra và điều chỉnh lại.
* **Phát triển theo từng đợt:**Sản phẩm có thể được phát hành từng phần, giúp người dùng sớm trải nghiệm và đánh giá tính năng, đồng thời cung cấp thông tin phản hồi quý báu cho các vòng lặp tiếp theo.
* **Tăng cường giao tiếp với khách hàng:**Việc phát hành phiên bản sớm và nhận phản hồi thường xuyên giúp khách hàng cảm thấy có sự tham gia vào quá trình phát triển và sản phẩm cuối cùng sẽ phù hợp hơn với nhu cầu thực tế.

**Nhược điểm:**

* **Yêu cầu quản lý phức tạp:**Mỗi vòng lặp cần được lên kế hoạch, theo dõi và đánh giá riêng, đòi hỏi sự quản lý chặt chẽ để đảm bảo tiến độ và chất lượng.
* **Khó dự đoán chi phí và thời gian:**Vì các yêu cầu và tính năng có thể thay đổi theo vòng lặp, nên việc ước lượng tổng chi phí và thời gian hoàn thiện dự án trở nên khó khăn hơn so với các mô hình tuyến tính.
* **Nguy cơ thay đổi yêu cầu liên tục:**Phản hồi liên tục có thể dẫn đến việc khách hàng yêu cầu thay đổi không ngừng, gây khó khăn trong việc giữ vững phạm vi ban đầu của dự án (scope creep).
* **Rủi ro tích hợp:**Nếu không có sự chuẩn hóa và kiểm thử chặt chẽ, các thành phần được phát triển theo từng vòng lặp có thể gặp vấn đề khi tích hợp lại với nhau.
* **Quá trình lặp lại có thể kéo dài:**Nếu không được kiểm soát tốt, việc lặp lại quá nhiều vòng có thể làm chậm tiến độ tổng thể của dự án và làm tăng chi phí phát triển.

## Câu 3.4:Vì sao mô hình CMM được sử dụng rộng rãi trong quản lý chất lượng phần mềm?

* **Cải thiện quy trình:**CMMI cung cấp một khung làm việc toàn diện giúp cải thiện các quy trình phát triển và bảo trì phần mềm, từ đó nâng cao chất lượng sản phẩm và dịch vụ.
* **Đánh giá khách quan:**Mô hình cho phép đánh giá mức độ trưởng thành của các quy trình kinh doanh một cách khách quan qua các chỉ số định lượng, từ đó giúp nhận diện rõ ràng các lĩnh vực cần cải thiện.
* **Tăng cường hiệu quả:**Áp dụng CMMI giúp tối ưu hóa nguồn lực, giảm thiểu lỗi và nâng cao hiệu suất làm việc, dẫn đến hiệu quả kinh doanh cao hơn và giảm chi phí phát triển phần mềm.
* **Đáp ứng yêu cầu khách hàng:**Với CMMI, các tổ chức có thể đảm bảo rằng sản phẩm và dịch vụ của họ không chỉ đáp ứng mà còn vượt qua kỳ vọng của khách hàng về chất lượng và độ tin cậy.
* **Tăng cường khả năng cạnh tranh:**Việc áp dụng CMMI cho thấy quy trình làm việc chuyên nghiệp và tiêu chuẩn cao, từ đó giúp tổ chức nổi bật so với các đối thủ không áp dụng mô hình này.
* **Hỗ trợ phát triển liên tục:**CMMI thúc đẩy văn hóa cải tiến không ngừng, giúp tổ chức luôn sẵn sàng đối mặt với những thay đổi và thách thức mới, duy trì khả năng phát triển lâu dài.

## Câu 3.5:Thảo luận về các khó khăn khi áp dụng mô hình CMM trong thực tế.

#### 1. Chi phí và thời gian cao

* Việc **chuẩn hóa quy trình, đào tạo nhân viên** và áp dụng CMM tốn nhiều **chi phí, thời gian**.
* Các công ty nhỏ hoặc startup thường không đủ ngân sách để triển khai đầy đủ.

#### 2. Chống đối từ nhân viên

* Nhân viên có thể cảm thấy **bị ràng buộc bởi quy trình** và không thoải mái với những thay đổi mới.
* Nếu quy trình phức tạp, lập trình viên có thể thấy mất tự do sáng tạo.

#### 3. Khó khăn trong việc đo lường và kiểm soát

* Ở **mức 4 và mức 5**, cần có hệ thống đo lường hiệu suất và kiểm soát chặt chẽ.
* Nếu không có công cụ hỗ trợ tốt, việc thu thập và phân tích dữ liệu sẽ gặp khó khăn.

#### 4. Khó tích hợp với phương pháp linh hoạt (Agile, Scrum)

* CMM phù hợp với các tổ chức lớn có quy trình chặt chẽ, nhưng **khó kết hợp với mô hình phát triển linh hoạt (Agile, Scrum)**.
* Các công ty phát triển phần mềm nhanh có thể thấy CMM quá cứng nhắc.

#### 5. Quá trình triển khai phức tạp

* Việc **định nghĩa quy trình, tài liệu hóa và thực hiện kiểm tra** có thể làm **tăng gánh nặng hành chính**.
* Nếu không có đội ngũ chuyên trách, việc duy trì mô hình CMM dễ thất bại.

## Câu 3.6:Giải pháp để cải tiến quy trình phát triển phần mềm.

#### 1. Kết hợp mô hình CMM với Agile/Scrum

* Thay vì áp dụng CMM một cách cứng nhắc, có thể **kết hợp với Agile** để tăng tính linh hoạt.
* Sử dụng **Scrum** để quản lý nhóm nhỏ, nhưng vẫn áp dụng **các tiêu chuẩn CMM** để đảm bảo chất lượng.

#### 2. Tự động hóa quy trình phát triển

* Sử dụng công cụ như **Jenkins, GitHub Actions** để tự động hóa kiểm thử và triển khai.
* Áp dụng **CI/CD (Continuous Integration/Continuous Deployment)** để giảm lỗi và tăng tốc độ phát triển.

#### 3. Đào tạo và thay đổi tư duy nhân viên

* Cung cấp **khóa đào tạo về CMM** để nhân viên hiểu rõ lợi ích của mô hình.
* Thay vì áp đặt, **khuyến khích nhân viên tham gia đóng góp vào việc cải tiến quy trình**.

#### 4. Định nghĩa quy trình đơn giản, dễ hiểu

* Chỉ nên áp dụng **các quy trình cần thiết**, tránh làm mọi thứ trở nên quá phức tạp.
* Sử dụng **tài liệu ngắn gọn, dễ hiểu** thay vì quy trình dài dòng khó áp dụng.

#### 5. Đánh giá và cải tiến liên tục

* Sử dụng phương pháp **PDCA (Plan - Do - Check - Act)** để cải tiến quy trình một cách liên tục.
* Thu thập phản hồi từ nhân viên và khách hàng để điều chỉnh quy trình phù hợp.

## Câu 3.7:Phân tích ưu điểm của việc áp dụng tiến trình thống nhất trong các dự án lớn.

Tiến trình Thống nhất (UP) là một phương pháp phát triển phần mềm theo **mô hình lặp và tăng trưởng**, phù hợp với các dự án lớn, phức tạp. Dưới đây là những ưu điểm chính khi áp dụng UP vào các dự án phần mềm quy mô lớn:

* **Giảm rủi ro thông qua phát triển lặp**

✅ Dự án được chia thành nhiều vòng lặp nhỏ, giúp phát hiện lỗi và điều chỉnh sớm, giảm nguy cơ thất bại.  
✅ Các tính năng quan trọng được phát triển và kiểm thử trước, đảm bảo sản phẩm luôn có tiến triển rõ ràng.

**Ví dụ:** Trong một dự án hệ thống ngân hàng, nếu phát hiện sai sót trong quản lý giao dịch sớm, nhóm phát triển có thể điều chỉnh trước khi toàn bộ hệ thống được triển khai.

* **Linh hoạt và thích nghi với thay đổi**

✅ Nếu yêu cầu phần mềm thay đổi trong quá trình phát triển, có thể cập nhật trong các vòng lặp tiếp theo.  
✅ Khách hàng có thể đưa ra phản hồi và yêu cầu cải tiến mà không cần đợi đến khi hoàn thành toàn bộ dự án.

**Ví dụ:** Trong một dự án thương mại điện tử, nếu khách hàng muốn thêm tính năng thanh toán mới, nhóm phát triển có thể tích hợp trong vòng lặp kế tiếp mà không ảnh hưởng đến tiến trình tổng thể.

* **Tập trung vào kiến trúc phần mềm ngay từ đầu**

✅ Giai đoạn **Lập kế hoạch (Elaboration)** giúp thiết kế một kiến trúc mạnh mẽ trước khi đi vào lập trình chi tiết.  
✅ Một kiến trúc tốt giúp hệ thống có khả năng mở rộng, bảo trì và nâng cấp dễ dàng.

**Ví dụ:** Hệ thống phần mềm quản lý doanh nghiệp lớn cần có kiến trúc vững chắc để hỗ trợ hàng nghìn người dùng mà không ảnh hưởng đến hiệu suất.

* **Kiểm thử và đảm bảo chất lượng liên tục**

✅ Mỗi vòng lặp đều có giai đoạn kiểm thử (test), giúp phát hiện lỗi sớm.  
✅ Cải thiện chất lượng sản phẩm ngay từ những giai đoạn đầu.

**Ví dụ:** Trong một dự án phần mềm y tế, việc kiểm thử sớm giúp đảm bảo hệ thống không gặp lỗi nghiêm trọng, tránh ảnh hưởng đến việc chăm sóc bệnh nhân.

* **Quản lý dự án hiệu quả hơn**

✅ Tiến trình UP chia dự án thành các **mốc rõ ràng** với các giai đoạn: **Khởi đầu → Lập kế hoạch → Xây dựng → Chuyển giao**.  
✅ Dễ dàng theo dõi tiến độ, kiểm soát ngân sách và tài nguyên.

**Ví dụ:** Một dự án phát triển phần mềm cho chính phủ cần có kế hoạch rõ ràng để đảm bảo tiến độ và tuân thủ quy định.

* **Tăng cường cộng tác giữa các bên liên quan**

✅ Khách hàng, nhà quản lý và nhóm phát triển có thể tham gia vào từng giai đoạn, giúp dự án đi đúng hướng.  
✅ Giảm thiểu hiểu lầm giữa nhóm kỹ thuật và người dùng cuối.

**Ví dụ:** Trong một dự án phần mềm quản lý bệnh viện, các bác sĩ có thể kiểm tra từng phiên bản nhỏ để đảm bảo hệ thống phù hợp với nhu cầu thực tế.

## Câu 3.8:Thảo luận về sự cần thiết của việc kiểm thử trong từng pha của tiến trình thống nhất.

**Tiến trình Thống nhất (Unified Process - UP)** áp dụng mô hình **lặp và tăng trưởng**, chia phát triển phần mềm thành 4 pha chính: **Khởi đầu (Inception), Lập kế hoạch (Elaboration), Xây dựng (Construction), Chuyển giao (Transition)**. Kiểm thử (Testing) được thực hiện xuyên suốt trong từng pha để đảm bảo **chất lượng phần mềm, giảm rủi ro và tăng độ tin cậy**.

Dưới đây là vai trò của kiểm thử trong từng pha:

* **Kiểm thử trong pha Khởi đầu (Inception)**

📌 **Mục tiêu:** Xác định phạm vi dự án, yêu cầu chính và rủi ro tiềm ẩn.

✅ **Kiểm thử được áp dụng như thế nào?**

* **Kiểm tra yêu cầu ban đầu**: Đảm bảo yêu cầu rõ ràng, không mâu thuẫn.
* **Phân tích rủi ro**: Xác định các yếu tố có thể ảnh hưởng đến chất lượng phần mềm.
* **Kiểm thử khả thi**: Đánh giá tính khả thi của công nghệ, nguồn lực và thời gian.

**🔹 Nếu không kiểm thử:** Yêu cầu sai hoặc không rõ ràng có thể dẫn đến phát triển sai hướng ngay từ đầu.

* **Kiểm thử trong pha Lập kế hoạch (Elaboration)**

📌 **Mục tiêu:** Xây dựng kiến trúc hệ thống và xác định các thành phần quan trọng.

✅ **Kiểm thử được áp dụng như thế nào?**

* **Kiểm thử nguyên mẫu (Prototype Testing):** Thử nghiệm bản mô phỏng giao diện hoặc một phần chức năng để xác minh tính khả thi.
* **Kiểm thử kiến trúc (Architecture Testing):** Đánh giá khả năng mở rộng, bảo mật và hiệu suất của kiến trúc hệ thống.
* **Kiểm thử yêu cầu (Requirement Validation):** Đảm bảo yêu cầu đầy đủ, tránh hiểu sai.

**🔹 Nếu không kiểm thử:** Kiến trúc sai hoặc không tối ưu sẽ làm phần mềm khó mở rộng và bảo trì về sau.

* **Kiểm thử trong pha Xây dựng (Construction)**

📌 **Mục tiêu:** Lập trình và phát triển các tính năng của phần mềm.

✅ **Kiểm thử được áp dụng như thế nào?**

* **Kiểm thử đơn vị (Unit Testing):** Kiểm tra từng thành phần nhỏ của phần mềm.
* **Kiểm thử tích hợp (Integration Testing):** Đảm bảo các module hoạt động đúng khi kết hợp với nhau.
* **Kiểm thử chức năng (Functional Testing):** Kiểm tra xem hệ thống có hoạt động đúng theo yêu cầu không.
* **Kiểm thử hiệu suất (Performance Testing):** Đánh giá tốc độ và khả năng chịu tải của hệ thống.

**🔹 Nếu không kiểm thử:** Lỗi có thể xuất hiện hàng loạt khi tích hợp, gây tốn kém thời gian sửa chữa.

* **Kiểm thử trong pha Chuyển giao (Transition)**

📌 **Mục tiêu:** Hoàn thiện, triển khai phần mềm và bàn giao cho khách hàng.

✅ **Kiểm thử được áp dụng như thế nào?**

* **Kiểm thử hệ thống (System Testing):** Đánh giá toàn bộ phần mềm trước khi triển khai.
* **Kiểm thử chấp nhận (User Acceptance Testing - UAT):** Người dùng thực hiện kiểm tra để xác nhận phần mềm đáp ứng nhu cầu.
* **Kiểm thử bảo mật (Security Testing):** Đảm bảo phần mềm an toàn trước các mối đe dọa.

**🔹 Nếu không kiểm thử:** Phần mềm có thể gặp lỗi nghiêm trọng khi triển khai, gây mất uy tín và tốn kém chi phí sửa chữa.

## Câu 3.9: So sánh giữa mô hình CMM mức 4 và mức 5.

* Mức 4 (Quản lý định lượng) tập trung kiểm soát quy trình bằng dữ liệu định lượng, đảm bảo ổn định, giảm sai sót và dự đoán kết quả.
* Mức 5 (Tối ưu hóa) nhấn mạnh cải tiến liên tục dựa trên dữ liệu và phản hồi, hướng tới sự hoàn hảo và linh hoạt.
* Khác biệt chính: Mức 4 ưu tiên quản lý ổn định, mức 5 chú trọng tối ưu hóa không ngừng và đổi mới.

## Câu 3.10: Đề xuất cách tổ chức hoạt động nhóm trong workflow lấy yêu cầu

* Phân vai trò: Chỉ định trưởng nhóm điều phối, chuyên viên phỏng vấn giao tiếp với khách hàng, chuyên viên phân tích ghi nhận yêu cầu, và người soạn thảo tài liệu SRS.
* Chuẩn bị phỏng vấn/khảo sát: Họp nhóm để xây dựng câu hỏi, phân công 2-3 người thực hiện phỏng vấn hoặc gửi khảo sát, thu thập thông tin đầy đủ.
* Phân tích yêu cầu: Tổ chức buổi làm việc nhóm, phân loại yêu cầu chức năng và phi chức năng.
* Soạn SRS: Nhóm lập tài liệu phối hợp với phân tích viên để viết SRS chi tiết, review nội bộ đảm bảo chất lượng trước khi gửi khách hàng.
* Duyệt với khách hàng: Trình bày SRS, chỉnh sửa theo phản hồi, hoàn thiện tài liệu được duyệt.

# PHẦN 4: CÂU HỎI TÌNH HUỐNG

## Câu 4.1: Một công ty phát triển phần mềm gặp khó khăn khi yêu cầu của khách hàng liên tục thay đổi trong pha xây dựng. Đội phát triển nên làm gì để giải quyết vấn đề này?

Khi yêu cầu khách hàng liên tục thay đổi trong pha xây dựng, đội phát triển nên:

1. Thiết lập quy trình quản lý thay đổi chính thức
2. Áp dụng phương pháp phát triển linh hoạt với các sprint ngắn
3. Phân tích tác động của thay đổi đến lịch trình và ngân sách
4. Tăng cường giao tiếp với khách hàng qua các cuộc họp thường xuyên
5. Sử dụng nguyên mẫu để xác nhận hiểu biết về yêu cầu
6. Ưu tiên hoàn thành các yêu cầu cốt lõi trước
7. Điều chỉnh hợp đồng và phạm vi nếu cần thiết
8. Tăng cường kiểm thử tự động và liên tục
9. Thiết kế hệ thống có kiến trúc mở, dễ thích ứng với thay đổi
10. Rà soát lại pha làm rõ (Elaboration) để tránh vấn đề tương tự trong tương lai

## Câu 4.2: Trong pha chuyển giao của tiến trình thống nhất, khách hàng yêu cầu bổ sung thêm tính năng mới. Đội phát triển nên xử lý ra

## sao?

Đội phát triển nên:

1. Đánh giá yêu cầu – Xác định mức độ quan trọng, phạm vi và tác động.
2. Xem xét tiến trình – Kiểm tra ảnh hưởng đến hệ thống và tiến độ bàn giao.
3. Trao đổi với khách hàng – Thống nhất về thời gian, chi phí và ưu tiên.
4. Triển khai nếu khả thi – Lập kế hoạch bổ sung hoặc đưa vào phiên bản sau.

## Câu 4.3:Dự án phát triển phần mềm bị trễ tiến độ do lỗi phát sinh liên tục trong quá trình kiểm thử. Là trưởng dự án, bạn sẽ làm gì?

**Xem xét lại quy trình kiểm thử:**

* Xác định nguyên nhân gốc rễ của các lỗi phát sinh và xem xét lại các bước kiểm thử hiện tại để đảm bảo rằng không có lỗi nào bị bỏ sót ở các giai đoạn trước đó.
* Sử dụng các công cụ kiểm thử tự động và phân tích mã nguồn để giúp phát hiện sớm các lỗi.

**Tăng cường giao tiếp:**

* Thiết lập các cuộc họp định kỳ giữa các nhóm phát triển, kiểm thử và quản lý dự án để đảm bảo rằng mọi người đều nắm bắt được tình hình.
* Chia sẻ báo cáo tiến độ, thảo luận về các vấn đề phát sinh và cùng nhau tìm giải pháp để khắc phục lỗi.

**Đào tạo và nâng cao kỹ năng:**

* Cung cấp đào tạo bổ sung cho đội ngũ kiểm thử và phát triển về các kỹ thuật, công cụ kiểm thử mới cũng như các công nghệ hiện đại.
* Tổ chức các buổi workshop chia sẻ kinh nghiệm nhằm nâng cao năng lực phát hiện và xử lý lỗi hiệu quả hơn.

**Tăng cường tài nguyên:**

* Xem xét bổ sung nhân lực hoặc tài nguyên cần thiết để giảm tải công việc cho các thành viên hiện tại, giúp xử lý lỗi nhanh hơn và hiệu quả hơn.

**Kiểm tra chất lượng mã nguồn:**

* Áp dụng các công cụ kiểm tra tự động (như static code analysis, continuous integration tools) để đánh giá chất lượng mã nguồn và phát hiện lỗi sớm trong quá trình phát triển.

**Ưu tiên xử lý lỗi:**

* Đặt ưu tiên cao cho việc xử lý các lỗi nghiêm trọng có ảnh hưởng trực tiếp đến tiến độ và chất lượng sản phẩm.
* Phân loại lỗi theo mức độ nghiêm trọng và ưu tiên sửa chữa những lỗi có tác động lớn nhất.

**Theo dõi và giám sát chặt chẽ:**

* Sử dụng các công cụ theo dõi lỗi và báo cáo tiến độ để giám sát tình hình của dự án một cách liên tục.
* Thiết lập hệ thống cảnh báo sớm để có thể phản ứng kịp thời nếu có vấn đề phát sinh mới.

**Tạo động lực và khuyến khích nhóm:**

* Ghi nhận và khen thưởng những nỗ lực, thành tích của các thành viên trong quá trình xử lý lỗi và đạt được các mốc quan trọng.
* Tạo môi trường làm việc tích cực để duy trì tinh thần và cam kết của toàn đội đối với dự án.

**Thay đổi phương pháp làm việc (nếu cần):**

* Xem xét chuyển sang các phương pháp làm việc linh hoạt như Agile hoặc Scrum để tăng tính phản hồi nhanh và khả năng thích ứng với các thay đổi phát sinh.
* Điều chỉnh quy trình nội bộ để phù hợp hơn với yêu cầu kiểm thử liên tục và phát triển theo từng sprint.

**Báo cáo và điều chỉnh kế hoạch:**

* Thiết lập quy trình báo cáo định kỳ để cập nhật tiến độ, các vấn đề phát sinh và kết quả xử lý lỗi cho ban quản lý dự án.
* Dựa trên các báo cáo này, phân tích dữ liệu và điều chỉnh kế hoạch dự án (bao gồm thời gian, nguồn lực và mục tiêu) nhằm đưa dự án trở lại đúng tiến độ và đảm bảo chất lượng sản phẩm.
* Đảm bảo các điều chỉnh được thực hiện linh hoạt và kịp thời để ứng phó với thay đổi của môi trường làm việc và yêu cầu của khách hàng.

## Câu 4.4:Trong workflow thiết kế, kiến trúc sư phần mềm muốn thay đổi thiết kế ban đầu để cải thiện hiệu suất. Đội phát triển nên xử lý thế nào?

**Tổ chức cuộc họp với các bên liên quan:**

* Tổ chức cuộc họp bao gồm kiến trúc sư, đội phát triển và quản lý dự án để thảo luận về đề xuất thay đổi.
* Mục tiêu là đảm bảo rằng tất cả các bên đều hiểu rõ lý do và mục tiêu của việc thay đổi, từ đó tạo ra sự đồng thuận và cam kết chung.

**Đánh giá tác động:**

* Phân tích chi tiết tác động của thay đổi đối với các phần khác của hệ thống và quy trình phát triển.
* Xác định các rủi ro tiềm ẩn (ví dụ: ảnh hưởng đến bảo mật, khả năng mở rộng hoặc tương thích với các module khác) và đưa ra các biện pháp giảm thiểu.

**Cập nhật tài liệu:**

* Khi thay đổi được chấp thuận, cập nhật các tài liệu thiết kế và tài liệu liên quan để phản ánh thiết kế mới.
* Việc này đảm bảo rằng toàn bộ thông tin được ghi chép chính xác và đầy đủ, hỗ trợ cho việc triển khai và bảo trì sau này.

**Lập kế hoạch thực hiện:**

* Xác định các bước cụ thể để thực hiện thay đổi, bao gồm phân công nhiệm vụ, xác định thời gian hoàn thành và lên lịch các cuộc kiểm tra.
* Lập kế hoạch chi tiết giúp đảm bảo rằng quá trình chuyển đổi diễn ra mượt mà, không làm gián đoạn các chức năng quan trọng của hệ thống.

**Thực hiện thay đổi:**

* Bắt đầu thực hiện thay đổi theo kế hoạch đã đề ra.
* Đảm bảo rằng mọi người trong đội phát triển đều tuân thủ các nguyên tắc và quy trình đã thống nhất, đồng thời xử lý kịp thời các vấn đề phát sinh.

**Kiểm thử kỹ lưỡng:**

* Thực hiện kiểm thử toàn diện (bao gồm kiểm thử chức năng, tích hợp và hiệu suất) để đảm bảo rằng thiết kế mới cải thiện hiệu suất như mong đợi mà không gây ra lỗi mới.
* Điều này giúp đảm bảo sản phẩm cuối cùng ổn định và đáp ứng các yêu cầu kỹ thuật.

**Triển khai và giám sát:**

* Triển khai thay đổi vào môi trường thực tế theo phương pháp từng bước (ví dụ: rolling update) để giảm thiểu rủi ro.
* Giám sát hệ thống sau triển khai để theo dõi hiệu suất, kiểm tra phản hồi từ người dùng và sẵn sàng điều chỉnh nếu có vấn đề phát sinh.

**Đánh giá và cải tiến liên tục:**

* Tổ chức các cuộc họp tổng kết (post-mortem) để đánh giá quá trình thay đổi, rút ra bài học kinh nghiệm và cải tiến quy trình làm việc trong tương lai.
* Việc này giúp nâng cao hiệu quả của quy trình phát triển và đảm bảo rằng các thay đổi tiếp theo sẽ được thực hiện một cách hiệu quả hơn.

## Câu 4.5:Khách hàng yêu cầu rút ngắn thời gian phát triển dự án mà không thay đổi yêu cầu. Đội phát triển nên phản ứng ra sao?

Khi khách hàng yêu cầu **rút ngắn thời gian phát triển mà không thay đổi yêu cầu**, đội phát triển cần xem xét cẩn thận để đảm bảo **chất lượng phần mềm** không bị ảnh hưởng. Các bước xử lý phù hợp gồm:

#### 1. Đánh giá lại kế hoạch dự án

* Xác định xem **đã tối ưu hóa quy trình chưa** (tự động hóa kiểm thử, dùng framework phù hợp, v.v.).
* Đánh giá xem **có thể cắt giảm công đoạn nào mà không ảnh hưởng đến chất lượng**.

#### 2. Đàm phán với khách hàng

* **Giải thích rủi ro**: Việc rút ngắn thời gian có thể dẫn đến lỗi phần mềm hoặc bỏ qua kiểm thử quan trọng.
* **Đề xuất phương án thay thế**, ví dụ:
  + Cung cấp **phiên bản phần mềm với các tính năng cốt lõi trước**, các tính năng khác bổ sung sau.
  + **Tăng tài nguyên (thêm người, tăng ca)** nhưng chi phí sẽ tăng.

#### 3. Huy động thêm nhân lực nếu cần

* Nếu khách hàng chấp nhận tăng ngân sách, có thể thuê thêm lập trình viên hoặc sử dụng outsourcing.
* Phân chia công việc hợp lý hơn để tăng tốc độ phát triển.

#### 4. Tăng cường tự động hóa

* Áp dụng **CI/CD, kiểm thử tự động (Selenium, JUnit, TestNG...)** để giảm thời gian kiểm thử thủ công.
* Sử dụng các **thư viện, framework có sẵn** thay vì viết mọi thứ từ đầu.

#### 5. Tập trung vào các phần quan trọng trước

* Ưu tiên **các tính năng cốt lõi** trước, các phần mở rộng có thể triển khai sau.
* Sử dụng phương pháp **phát triển theo giai đoạn (Incremental Development)**.

## Câu 4.6:Một công ty nhỏ muốn áp dụng mô hình CMM nhưng gặp khó khăn do thiếu nguồn lực. Hãy đề xuất giải pháp.

Các công ty nhỏ thường gặp khó khăn khi áp dụng **mô hình CMM** do hạn chế về nhân lực, thời gian và ngân sách. Một số giải pháp có thể áp dụng gồm:

#### 1. Bắt đầu từ mức CMM thấp và nâng dần

* Không cần phải đạt **CMM mức 5 ngay lập tức**, có thể bắt đầu từ **mức 2 (Managed)**.
* Định nghĩa **các quy trình cơ bản** như quản lý yêu cầu, kiểm thử phần mềm, quản lý lỗi.

#### 2. Tinh giản quy trình để phù hợp với công ty nhỏ

* Không áp dụng tất cả tiêu chuẩn CMM ngay lập tức, chỉ chọn **những phần quan trọng nhất**.
* Tránh **tài liệu hóa quá mức**, chỉ ghi chép những quy trình quan trọng.

#### 3. Sử dụng công cụ hỗ trợ để giảm tải công việc

* Dùng **Jira, Trello, ClickUp** để quản lý dự án.
* Sử dụng **GitHub/GitLab** để quản lý mã nguồn và theo dõi thay đổi.
* Áp dụng **kiểm thử tự động** để giảm tải kiểm thử thủ công.

#### 4. Đào tạo nhân viên theo từng bước

* Không cần đào tạo toàn bộ nhân viên cùng lúc, có thể **bắt đầu với nhóm quản lý và key members**.
* Cung cấp tài liệu, workshop nội bộ để nâng cao nhận thức về CMM.

#### 5. Tận dụng outsourcing hoặc tư vấn bên ngoài

* Nếu thiếu nguồn lực nội bộ, có thể thuê **chuyên gia tư vấn CMM** để hỗ trợ triển khai từng bước.
* Có thể thuê freelancer hoặc **outsource một số công việc** để giảm tải cho đội ngũ chính.

## Câu 4.7:Trong workflow lấy yêu cầu, khách hàng cung cấp thông tin không rõ ràng. Đội phát triển cần làm gì?

Khi khách hàng đưa ra thông tin **mơ hồ, không đầy đủ hoặc không nhất quán**, đội phát triển có thể gặp khó khăn trong việc thiết kế và lập trình phần mềm đúng với nhu cầu thực tế. Để giải quyết vấn đề này, nhóm phát triển có thể thực hiện các bước sau:

* **Làm rõ yêu cầu bằng cách đặt câu hỏi chi tiết**

📌 **Mục tiêu:** Xác định chính xác nhu cầu của khách hàng và tránh hiểu sai.

✅ **Hành động:**

* **Hỏi khách hàng về mục tiêu chính** của phần mềm: "Anh/chị mong đợi phần mềm sẽ giải quyết vấn đề gì?"
* **Sử dụng câu hỏi cụ thể thay vì chung chung**, ví dụ:
  + "Người dùng có thể thực hiện những thao tác nào trên hệ thống?"
  + "Hệ thống có cần phân quyền người dùng không?"
  + "Dữ liệu đầu vào và đầu ra của hệ thống là gì?"

**🔹 Nếu không làm rõ ngay từ đầu:** Nhóm phát triển có thể hiểu sai yêu cầu, dẫn đến sản phẩm không phù hợp.

* **Sử dụng phương pháp phân tích yêu cầu**

📌 **Mục tiêu:** Trực quan hóa yêu cầu để dễ hiểu và dễ kiểm tra.

✅ **Hành động:**

* **Vẽ sơ đồ Use Case** để mô tả các chức năng phần mềm từ góc nhìn người dùng.
* **Tạo User Stories** (câu chuyện người dùng) để mô tả hành vi của hệ thống dựa trên trải nghiệm thực tế.
* **Lập mô hình dữ liệu hoặc quy trình** (ví dụ: sơ đồ luồng dữ liệu DFD) để thể hiện cách hệ thống hoạt động.

**🔹 Nếu không áp dụng phương pháp phân tích:** Yêu cầu có thể bị hiểu sai và gây khó khăn khi phát triển.

* **Xây dựng prototype hoặc mô hình mẫu**

📌 **Mục tiêu:** Giúp khách hàng hình dung rõ về sản phẩm trước khi phát triển thực tế.

✅ **Hành động:**

* **Tạo bản thiết kế giao diện (wireframe, mockup)** để khách hàng xem và phản hồi.
* **Phát triển prototype** với một số chức năng cơ bản để khách hàng thử nghiệm.
* **Lấy phản hồi sớm** để điều chỉnh yêu cầu phù hợp hơn.

**🔹 Nếu không làm:** Dự án có thể đi sai hướng, gây tốn kém thời gian và chi phí sửa đổi sau này.

* **Ghi nhận và xác nhận yêu cầu với khách hàng**

📌 **Mục tiêu:** Đảm bảo rằng cả hai bên đều thống nhất về yêu cầu phần mềm.

✅ **Hành động:**

* **Lập tài liệu yêu cầu (SRS - Software Requirement Specification)** và gửi cho khách hàng kiểm tra.
* **Tổ chức cuộc họp xác nhận yêu cầu** với khách hàng trước khi bắt đầu phát triển.
* **Ký kết biên bản chốt yêu cầu** để tránh thay đổi liên tục sau này.

**🔹 Nếu không xác nhận yêu cầu:** Khách hàng có thể thay đổi ý kiến sau khi phát triển xong, gây mất thời gian.

* **Giữ liên lạc thường xuyên với khách hàng**

📌 **Mục tiêu:** Đảm bảo khách hàng có thể bổ sung hoặc điều chỉnh yêu cầu nếu cần.

✅ **Hành động:**

* **Tổ chức các cuộc họp định kỳ (weekly meeting, sprint review)** để cập nhật tiến độ và xác minh yêu cầu.
* **Dùng công cụ quản lý yêu cầu** (Jira, Trello, Notion) để khách hàng theo dõi tiến độ và phản hồi ngay khi cần.
* **Luôn có kênh giao tiếp mở** (email, chat, video call) để xử lý các thay đổi kịp thời.

**🔹 Nếu không duy trì liên lạc:** Khách hàng có thể không hài lòng với sản phẩm cuối cùng vì không đúng mong muốn.

## Câu 4.8: Một dự án gặp rủi ro cao trong pha khởi đầu do thiếu tài liệu yêu cầu rõ ràng. Đội phát triển nên làm gì?

Trong pha **Khởi đầu (Inception)** của một dự án phần mềm, nếu **tài liệu yêu cầu không rõ ràng**, nhóm phát triển có thể gặp các rủi ro như:

* **Hiểu sai yêu cầu**, dẫn đến phát triển sai hướng.
* **Tốn nhiều chi phí và thời gian sửa đổi** sau này.
* **Khách hàng không hài lòng** với sản phẩm cuối cùng.

Để giảm rủi ro, nhóm phát triển nên thực hiện các bước sau:

* **Chủ động làm rõ yêu cầu với khách hàng**

📌 **Mục tiêu:** Xác định đúng nhu cầu thực tế và mong đợi của khách hàng.

✅ **Hành động:**

* **Tổ chức các buổi họp phân tích yêu cầu** với khách hàng.
* **Hỏi khách hàng bằng câu hỏi cụ thể**, ví dụ:
  + "Những tính năng nào là quan trọng nhất?"
  + "Ai là người sử dụng hệ thống?"
  + "Kết quả mong muốn của hệ thống là gì?"
* **Dùng kỹ thuật phỏng vấn, khảo sát** để thu thập thêm thông tin từ khách hàng và người dùng cuối.

🔹 **Nếu không làm rõ ngay từ đầu:** Nhóm phát triển có thể đi sai hướng, gây lãng phí thời gian và công sức.

* **Xây dựng tài liệu yêu cầu ban đầu**

📌 **Mục tiêu:** Định nghĩa rõ ràng yêu cầu để làm cơ sở cho thiết kế và phát triển.

✅ **Hành động:**

* **Viết tài liệu SRS (Software Requirement Specification)** mô tả chi tiết các yêu cầu phần mềm.
* **Tạo User Stories** (câu chuyện người dùng) để mô tả cách hệ thống hoạt động.
* **Xác định các chức năng chính và ưu tiên** theo mức độ quan trọng.

🔹 **Nếu không có tài liệu yêu cầu:** Đội phát triển dễ hiểu sai và mất thời gian sửa lỗi về sau.

* **Sử dụng mô hình trực quan để làm rõ yêu cầu**

📌 **Mục tiêu:** Giúp khách hàng dễ hiểu hơn về hệ thống.

✅ **Hành động:**

* **Vẽ sơ đồ Use Case** để mô tả cách người dùng tương tác với hệ thống.
* **Tạo sơ đồ luồng dữ liệu (DFD)** để minh họa cách dữ liệu di chuyển trong hệ thống.
* **Thiết kế giao diện mẫu (Wireframe, Mockup)** để khách hàng hình dung sản phẩm.

🔹 **Nếu không trực quan hóa yêu cầu:** Khách hàng có thể khó hiểu và đưa ra yêu cầu mơ hồ.

* **Phát triển nguyên mẫu (Prototype) để khách hàng xác nhận**

📌 **Mục tiêu:** Kiểm tra tính khả thi của yêu cầu trước khi phát triển chính thức.

✅ **Hành động:**

* **Xây dựng một nguyên mẫu đơn giản** với một số tính năng chính.
* **Cho khách hàng thử nghiệm sớm** để thu thập phản hồi.
* **Điều chỉnh yêu cầu dựa trên phản hồi của khách hàng**.

🔹 **Nếu không có prototype:** Nhóm phát triển có thể xây dựng sai chức năng mong muốn.

* **Áp dụng phương pháp phát triển linh hoạt (Agile, Scrum)**

📌 **Mục tiêu:** Giúp điều chỉnh yêu cầu linh hoạt nếu có thay đổi.

✅ **Hành động:**

* **Chia dự án thành các vòng lặp nhỏ (Sprint)** thay vì phát triển toàn bộ một lúc.
* **Thường xuyên họp phản hồi với khách hàng** để điều chỉnh kịp thời.
* **Ưu tiên phát triển các tính năng cốt lõi trước**, sau đó mở rộng dần.

🔹 **Nếu không áp dụng Agile:** Việc thay đổi yêu cầu sau này có thể gây gián đoạn lớn.

* **Xác nhận yêu cầu chính thức với khách hàng**

📌 **Mục tiêu:** Đảm bảo tất cả các bên đều thống nhất về yêu cầu phần mềm.

✅ **Hành động:**

* **Gửi tài liệu yêu cầu cho khách hàng kiểm tra và xác nhận**.
* **Tạo biên bản thỏa thuận yêu cầu** để tránh tranh cãi về sau.
* **Định nghĩa rõ ràng phạm vi dự án** để tránh yêu cầu mở rộng ngoài kế hoạch.

🔹 **Nếu không xác nhận yêu cầu:** Dự án dễ bị thay đổi liên tục, dẫn đến chi phí phát triển cao.

## Câu 4.9: Dự án phần mềm lớn có nhiều nhóm phát triển ở các địa điểm khác nhau. Làm thế nào để đảm bảo các nhóm phối hợp hiệu quả?

Để đảm bảo các nhóm phát triển phần mềm ở nhiều địa điểm khác nhau phối hợp hiệu quả trong một dự án lớn, có thể thực hiện theo các cách sau.

* Trước hết, về giao tiếp, cần thiết lập các kênh liên lạc chung như ứng dụng chat (Microsoft Teams), email và họp trực tuyến qua các nền tảng như Zoom hay Google Meet, đồng thời tổ chức các cuộc họp định kỳ, chẳng hạn họp hàng ngày kiểu "stand-up" hoặc hàng tuần, để cập nhật tiến độ và giải quyết vấn đề.
* Thứ hai, về công cụ và quy trình, nên sử dụng các hệ thống quản lý dự án như Jira, Trello hoặc Asana để phân công nhiệm vụ và theo dõi tiến độ, kết hợp với quản lý mã nguồn qua Git trên các nền tảng như GitHub hay GitLab để đồng bộ hóa code, đồng thời tài liệu hóa mọi thông tin và thống nhất quy trình làm việc để các nhóm hoạt động nhịp nhàng.

## Câu 4.10: Một công ty phát triển phần mềm gặp khó khăn trong việc quản lý quy trình do không có chuẩn hóa. Hãy đề xuất giải pháp.

Để giải quyết khó khăn trong việc quản lý quy trình do không có chuẩn hóa, công ty cần:

* Thống nhất quy trình: Chọn phương pháp (Agile, Scrum), áp dụng đồng bộ qua các giai đoạn phát triển.
* Chuẩn hóa công cụ: Dùng Jira, Git, Confluence; lưu trữ tài liệu tập trung, rõ ràng.
* Đặt tiêu chuẩn: Quy tắc code, kiểm thử, code review; giám sát chất lượng.
* Đào tạo & theo dõi: Hướng dẫn nhân viên, kiểm tra định kỳ, điều chỉnh kịp thời.

## 

## 