HƯỚNG DẪN ÔN TẬP

HỌC PHẦN QUẢN LÝ DỰ ÁN PHẦN MỀM

Mục Lục

Lý thuyết (6.0 điểm), 02 CÂU

Phân tích các giai đoạn của quản lý dự án phần mềm

1. Khởi tạo dự án (Initiating)

Trước khi khởi tạo, doanh nghiệp cần đưa ra mục tiêu, lý do (business case) thiết lập và phát triển dự án. Trong giai đoạn này, người quản lý cần xử lý một số công việc sau:

- Xác định mục đích phát triển dự án phần mềm đối với doanh nghiệp và khách hàng. Đồng thời liệt kê những lợi ích và lượng dữ liệu được cung cấp bởi phần mềm đó.
- Phát triển quy tắc dự án (project charter) để xác định quyền hạn và vị trí thực tế của bản thân đối với dự án. Đây là cơ sở để bạn có tiếng nói hơn khi điều hành dự án phần mềm hoặc ứng dụng.
- Liệt kê các bên liên quan được hưởng lợi từ dự án để xác định phạm vi thực hiện.
- Đánh giá văn hóa và quy trình làm việc hiện tại của doanh nghiệp có phù hợp với mục tiêu của dự án hay không. Nếu không thì tiến hành đào tạo, cải tổ và tìm kiếm nguồn nhân lực phù hợp cho dự án. Những người đó có thể là: chuyên viên thiết kế, lập trình, kiểm thử, cài đặt, chăm sóc khách hàng…
- Chia dự án thành các phiên nhỏ để phân công nhiệm vụ phù hợp cho từng nhân viên và phòng ban liên quan.

- Nghiên cứu và thu thập các trình dữ liệu của hiện tại và quá khứ có liên quan đến dự án cần khởi tạo để đưa ra giả định về các vấn đề như: rủi ro, ràng buộc, thỏa thuận…
- Tìm hiểu những hạn chế và tính khả thi của phần mềm đối với nhu cầu thị trường, cuộc sống,…

2. Lập kế hoạch dự án (Planning)

Kế hoạch quản lý dự án phần mềm và ứng dụng cần phản ánh xuyên suốt quá trình thực hiện cho đến khi bàn giao. Thông qua plan có sẵn, người quản lý mới nắm vững đầu việc cũng như các khoản ngân sách thực tế cho quá trình chạy dự án. Người quản lý nên lập kế hoạch cụ thể cho từng giai đoạn phát triển phần mềm như: kế hoạch chất lượng, thẩm định, cấu hình ứng dụng, bảo trì và kế hoạch phát triển đội ngũ.

Dưới đây là công việc chi tiết:

- Xác định các yêu cầu đối với dự án như: ngân sách thực hiện, số lượng nhân sự, mốc thời gian…
- Phác thảo phạm vi và tham số của dự án như: quy mô, nguồn lực, mức độ phức tạp của phần mềm…
- Chia nhỏ lịch trình và tạo task công việc cần thực hiện để dễ dàng hơn trong việc quản lý, theo dõi và đốc thúc nhân sự.
- Thiết lập chỉ số đo lường tiến độ và chất lượng cho từng task công việc cụ thể.
- Lên kế hoạch công việc cần thực hiện với các bên liên quan đến dự án.
- Xuyên suốt dự án, quy trình lập kế hoạch (plan) sẽ lặp đi lặp lại các công việc sau: lên lịch cho từng công việc => tiến hành theo kế hoạch => theo dõi tiến độ => đánh giá tham số => lên lịch cho tham số mới hoặc tham số chậm tiến độ => thỏa thuận deadline cho từng sản phẩm => tìm kiếm giải pháp cho vấn đề phát sinh.

3. Triển khai (Executing)

Trong giai đoạn triển khai, người quản lý cần chú trọng đến các vấn đề liên quan đến tiến độ, con người và báo cáo. Những công việc cụ thể đó bao gồm:

- Lập kế hoạch cụ thể cho từng giai đoạn hoặc task cụ thể để định hướng công việc cho đồng nghiệp, cấp dưới.
- Tổ chức và phân công nhiệm vụ cụ thể cho từng đội nhóm hoặc nhân viên theo kế hoạch định sẵn.
- Đốc thúc tiến độ, theo dõi deadline và thu thập dữ liệu cho từng phiên dự án để xác định hiệu suất.
- Phân tích và xác định các rủi ro để kịp thời đưa ra thay đổi nhằm đảm bảo chất lượng cho ứng dụng mà vẫn đảm bảo tiến độ.
- Yêu cầu, theo dõi các tài nguyên phần cứng và phần mềm để kiểm thử chất lượng ban đầu.

- Xuyên suốt quá trình thực hiện, người quản lý cần theo dõi sát sao kết quả làm việc của cả nhóm hoặc phòng ban. Đồng thời, đứng ra giải quyết các xung đột phát sinh của đội nhóm. Quá trình này nên kèm theo khen thưởng đối với nhân viên xuất sắc và phê bình những người làm việc chưa tốt.
- Viết báo cáo tiến độ cho cấp trên và các bên liên quan. Trong báo cáo nên nêu rõ vấn đề phát sinh (nếu có) để đề xuất phương án giải quyết với cấp trên. Các vấn đề này có thể là: ngân sách, nhân sự, kiến thức,…

4. Giám sát và kiểm soát (Monitoring & Control)

Giám sát và kiểm soát là giai đoạn đánh dấu dự án sắp sửa cán đích. Thông qua giám sát và kiểm soát người quản lý có thể đưa ra các chỉnh sửa kịp thời để điều hướng, duy trì tiến độ dự án. Những công việc cụ thể bao gồm:

- Đo lường hiệu suất thực tế với chỉ số trong kế hoạch quản lý để phân tích, đánh giá hiệu suất công việc.
- Ghi chú tất cả các thay đổi phát sinh trong suốt dự án cũng như nguyên nhân gây ra và các đề xuất thay đổi đã thực hiện.
- Chỉnh sửa plan quản lý theo từng task một cách chi tiết và cụ thể.
- Xử lý những đề xuất thay đổi từ cấp dưới. Những chấp thuận và từ chối đó là gì?
- Giám sát tương tác của các bên liên quan và thu thập ý kiến từ họ để có những sửa đổi nếu cần thiết.
- Tiếp tục đánh giá rủi ro cho các vấn đề như: kỹ thuật, chức năng, đầu ra… cho ứng dụng. Đề xuất phương án xử lý cho từng vấn đề.
- Kiểm thử phần mềm và ứng dụng thường xuyên để đảm bảo chất lượng cho sản phẩm cuối.
- Quản lý và kiểm soát tất cả các thủ tục dùng thử hoặc mua bán sản phẩm phần mềm.
- Ngoài ra, ở giai đoạn này người quản lý cần kiểm soát các vấn đề về nâng cấp chất lượng nhưng lại gây tốn kém chi phí và thời gian. Nếu phát sinh vấn đề trên, bạn nên tham khảo ý kiến của cấp trên hoặc không nên thực hiện nó để không đi chệch hướng của kế hoạch ban đầu.

5. Kết thúc (Closing)

Ở giai đoạn kết thúc, công việc của quản lý là vận hành và hoàn tất các thủ tục còn lại để tiếp thị ứng dụng đến thị trường. Những đầu việc của giai đoạn Closing gồm có:

- Hoàn tất các vấn đề: kiểm thử chất lượng, cho khách hàng dùng thử, đối soát kỹ thuật…
- Thu thập chấp nhận sản phẩm từ đồng nghiệp, cấp trên và các bên liên quan.
- Bàn giao sản phẩm đã hoàn thành và tiến hành update sản phẩm lên nền tảng phù hợp.
- Thu thập phản hồi, đánh giá của khách hàng về ứng dụng để tiếp tục điều chỉnh nếu cần thiết.

- Báo cáo hiệu suất vận hành của ứng dụng, phần mềm và ghi chú các thành tựu đã đạt được.
- Rút kinh nghiệm từ dự án đi trước để có những sửa đổi và bổ sung cho kế hoạch tiếp theo.

Lựa chọn mô hình phát triển phần mềm

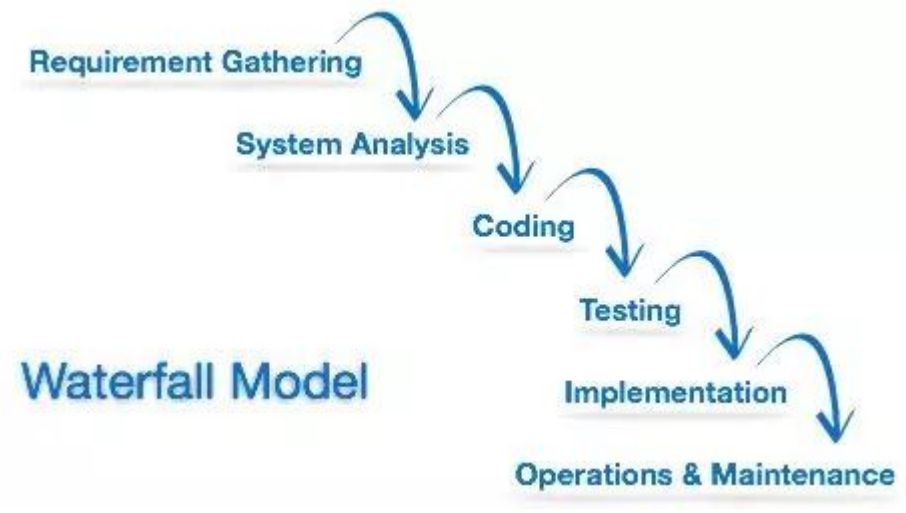
1. Định nghĩa

Mô hình phát triển phần mềm hay quy trình phát triển phần mềm xác định các pha/ giai đoạn trong xây dựng phần mềm. Có nhiều loại mô hình phát triển phần mềm khác nhau ví dụ như:

- Mô hình thác nước (Waterfall model)
- Mô hình xoắn ốc (Spiral model)
- Mô hình agile
- Mô hình tiếp cận lặp (Iterative model)
- Mô hình tăng trưởng (Incremental model)
- Mô hình chữ V (V model)
- Mô hình Scrum
- RAD model (Rapid Application Development)

Sau đây mình sẽ đi vào phân tích chi tiết từng mô hình.

1.1 Mô hình thác nước (Waterfall model)



Mô tả

- Đây được coi như là mô hình phát triển phần mềm đầu tiên được sử dụng.
- Mô hình này áp dụng tuần tự các giai đoạn của phát triển phần mềm.

- Đầu ra của giai đoạn trước là đầu vào của giai đoạn sau. Giai đoạn sau chỉ được thực hiện khi giai đoạn trước đã kết thúc. Đặc biệt không được quay lại giai đoạn trước để xử lý các yêu cầu khi muốn thay đổi.

Phân tích mô hình

- **Requirement gathering**: Thu thập và phân tích yêu cầu được ghi lại vào tài liệu đặc tả yêu cầu trong giai đoạn này.
- **System Analysis**: Phân tích thiết kế hệ thống phần mềm, xác định kiến trúc hệ thống tổng thể của phần mềm.
- **Coding**: Hệ thống được phát triển theo từng unit và được tích hợp trong giai đoạn tiếp theo. Mỗi Unit được phát triển và kiểm thử bởi dev được gọi là Unit Test.
- **Testing**: Cài đặt và kiểm thử phần mềm. Công việc chính của giai đoạn này là kiểm tra và sửa tất cả những lỗi tìm được sao cho phần mềm hoạt động chính xác và đúng theo tài liệu đặc tả yêu cầu.
- **Implementation**: Triển khai hệ thống trong môi trường khách hàng và đưa ra thị trường.
- **Operations and Maintenance**: Bảo trì hệ thống khi có bất kỳ thay đổi nào từ phía khách hàng, người sử dụng.

Ứng dụng

Mô hình thường được áp dụng cho các dự án phần mềm như sau:

- Các dự án nhỏ , ngắn hạn.
- Các dự án có ít thay đổi về yêu cầu và không có những yêu cầu không rõ ràng.

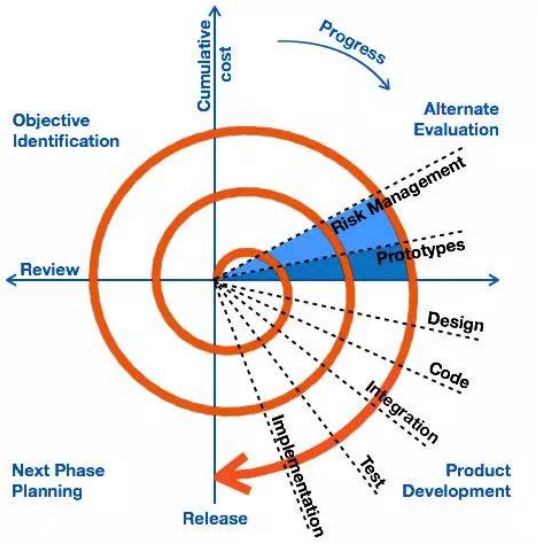
Ưu điểm

- Dễ sử dụng, dễ tiếp cận, dễ quản lý.
- Sản phẩm phát triển theo các giai đoạn được xác định rõ ràng.
- Xác nhận ở từng giai đoạn, đảm bảo phát hiện sớm các lỗi.

Nhược điểm

- Ít linh hoạt, phạm vi điều chỉnh hạn chế.
- Rất khó để đo lường sự phát triển trong từng giai đoạn.
- Mô hình không thích hợp với những dự án dài, đang diễn ra, hay những dự án phức tạp, có nhiều thay đổi về yêu cầu trong vòng đời phát triển.
- Khó quay lại khi giai đoạn nào đó đã kết thúc.

1.2 Mô hình xoắn ốc



Mô tả

- Là mô hình kết hợp giữa các tính năng của mô hình prototyping và mô hình thác nước.
- Mô hình xoắn ốc được ưa chuộng cho các dự án lớn, đắt tiền và phức tạp.
- Mô hình này sử dụng những giai đoạn tương tự như mô hình thác nước, về thứ tự, plan, đánh giá rủi ro, …

Phân tích mô hình

Các pha trong quy trình phát triển xoắn ốc bao gồm:

- **Objective identification- Thiết lập mục tiêu**: xác định mục tiêu, đối tượng cho từng pha của dự án.
- **Alternate evaluation- Đánh giá và giảm thiểu rủi ro**: đánh giá rủi ro và thực hiện các hành động để giảm thiểu rủi ro.
- **Product development- Phát triển sản phẩm**: Lựa chọn mô hình phù hợp để phát triển hệ thống.
- **Next phase planning- Lập kế hoạch**: đánh giá dự án và lập kế hoạch cho pha tiếp theo.

Ứng dụng

Mô hình này thường được sử dụng cho các ứng dụng lớn và các hệ thống được xây dựng theo các giai đoạn nhỏ hoặc theo các phân đoạn.

Ưu điểm

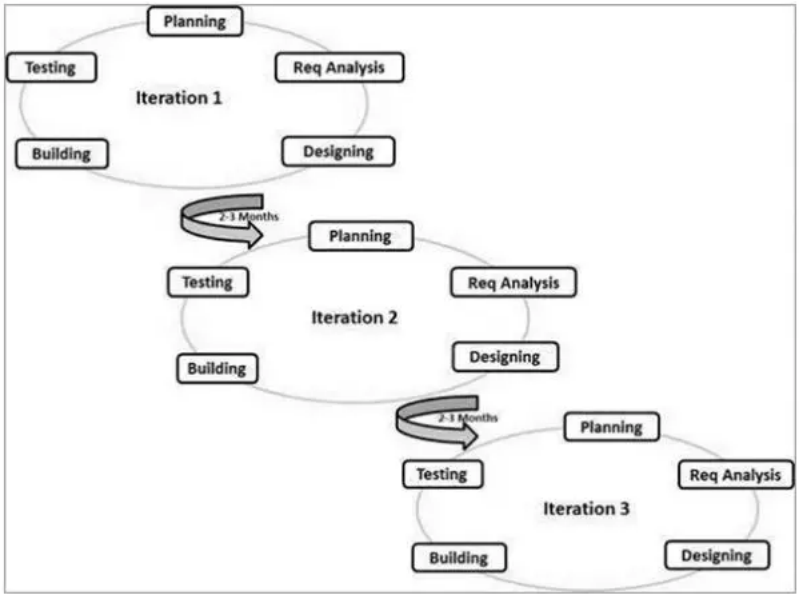
- Tốt cho các hệ phần mềm quy mô lớn.
- Dễ kiểm soát các mạo hiểm ở từng mức tiến hóa.

- Đánh giá thực tế hơn như là một quy trình làm việc, bởi vì những vấn đề quan trọng đã được phát hiện sớm hơn.

Nhược điểm

- Manager cần có kỹ năng tốt để quản lý dự án, đánh giá rủi ro kịp thời.
- Chi phí cao và mất nhiều thời gian để hoàn thành dự án.
- Phức tạp và không thích hợp với các dự án nhỏ và ít rủi ro.
- Yêu cầu thay đổi thường xuyên dẫn đến lặp vô hạn.
- Chưa được dùng rộng rãi.

1.3 Mô hình Agile



Agile là một phương pháp phát triển phần mềm linh hoạt để làm sao đưa sản phẩm đến tay người dùng càng nhanh càng tốt và được xem như là sự cải tiến so với những mô hình cũ như mô hình "Thác nước (waterfall)" hay "CMMI". Phương thức phát triển phần mềm Agile là một tập hợp các phương thức phát triển lặp và tăng dần trong đó các yêu cầu và giải pháp được phát triển

thông qua sự liên kết cộng tác giữa các nhóm tự quản và liên chức năng.

Mô tả

- Dựa trên mô hình iterative and incremental.
- Các yêu cầu và giải pháp phát triển dựa trên sự kết hợp của các function.
- Trong Agile, các tác vụ được chia thành các khung thời gian nhỏ để cung cấp các tính năng cụ thể cho bản phát hành cuối.

Ứng dụng

- Có thể được sử dụng với bất kỳ loại hình dự án nào, nhưng cần sự tham gia và tính tương tác của khách hàng.
- Sử dụng khi khách hàng yêu cầu chức năng sẵn sàng trong khoảng thời gian ngắn.

Ưu điểm

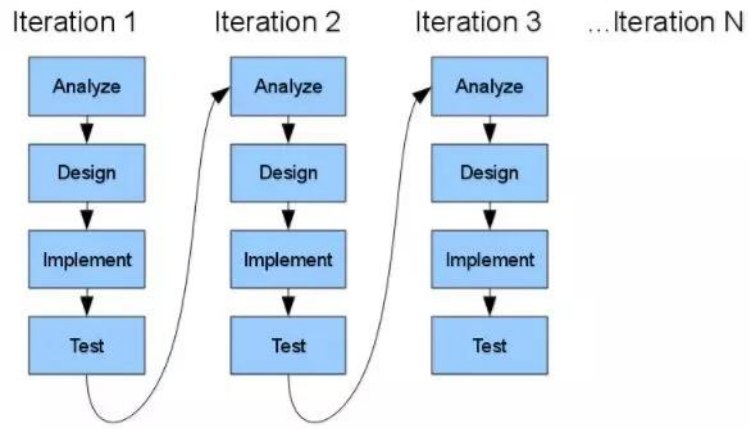
- Tăng cường tình thần làm việc nhóm và trao đổi công việc hiệu quả.
- Các chức năng được xây dựng nhanh chóng và rõ ràng, dế quản lý.
- Dễ dàng bổ sung, thay đổi yêu cầu.
- Quy tắc tối thiểu, tài liệu dễ hiểu, dễ sử dụng.

Nhược điểm

Mô hình Agile được sử dụng rộng rãi trên thế giới nhưng cũng không đồng nghĩa với phù hợp với tất cả các dự án phần mềm.

- Không thích hợp để xử lý các phụ thuộc phức tạp.
- Có nhiều rủi ro về tính bền vững, khả năng bảo trì và khả năng mở rộng.
- Cần một team có kinh nghiệm.
- Phụ thuộc rất nhiều vào sự tương tác rõ ràng của khách hàng.
- Chuyển giao công nghệ cho các thành viên mới trong nhóm có thể khá khó khăn do thiếu tài liệu.

1.4 Mô hình tiếp cận lặp



Mô tả

- Một mô hình được lặp đi lặp lại từ khi start cho đến khi làm đầy đủ spec.Quá trình này sau đó được lặp lại, tạo ra một phiên bản mới của phần mềm vào cuối mỗi lần lặp của mô hình.
- Thay vì phát triển phần mềm từ spec đặc tả rồi mới bắt đầu thực thi thì mô hình này có thể review dần dần để đi đến yêu cầu cuối cùng.

Ứng dụng

- Yêu cầu chính phải được xác định; tuy nhiên, một số chức năng hoặc yêu cầu cải tiến có thể phát triển theo thời gian.
- Một công nghệ mới đang được sử dụng và đang được học tập bởi nhóm phát triển trong khi làm việc trong dự án.
- Phù hợp cho các dự án lớn và nhiệm vụ quan trọng.

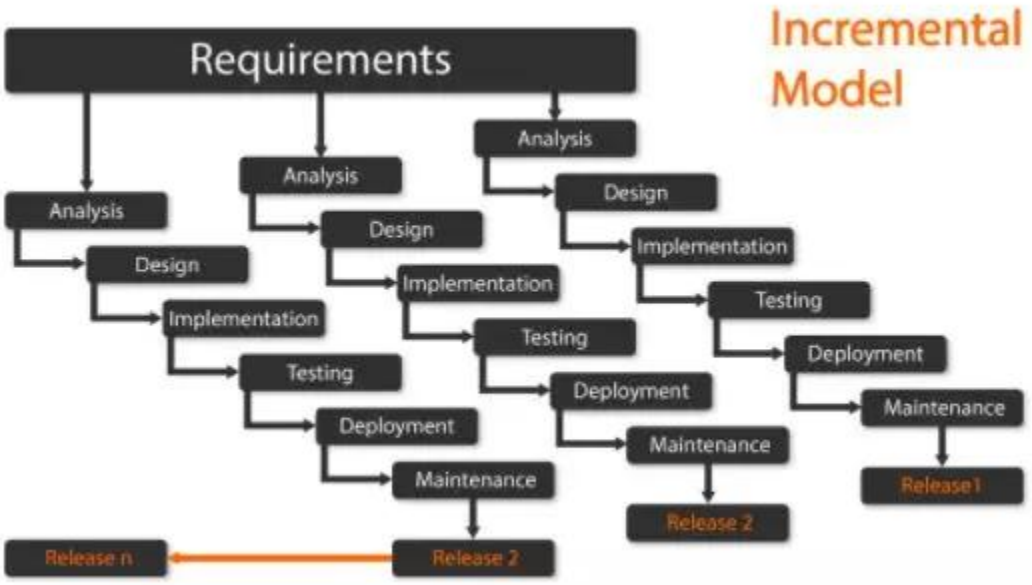
Ưu điểm

- Xây dựng và hoàn thiện các bước sản phẩm theo từng bước.
- Thời gian làm tài liệu sẽ ít hơn so với thời gian thiết kế.
- Một số chức năng làm việc có thể được phát triển nhanh chóng và sớm trong vòng đời.
- Ít tốn kém hơn khi thay đổ phạm vi, yêu cầu.
- Dễ quản lý rủi ro.
- Trong suốt vòng đời, phần mềm được sản xuất sớm để tạo điều kiện cho khách hàng đánh giá và phản hồi.

Nhược điểm

- Yếu cầu tài nguyên nhiều.
- Các vấn đề về thiết kế hoặc kiến trúc hệ thống có thể phát sinh bất cứ lúc nào.
- Yêu cầu quản lý phức tạp hơn.
- Tiến độ của dự án phụ thuộc nhiều vào giai đoạn phân tích rủi ro.

1.5 Mô hình tăng trưởng



Mô tả

- Spec được chia thành nhiều phần.
- Chu kỳ được chia thành các module nhỏ, dễ quản lý.
- Mỗi module sẽ đi qua các yêu cầu về thiết kế, thực hiện, … như 1 vòng đời phát triển thông thường.

Ứng dụng

- Áp dụng cho những dự án có yêu cầu đã được mô tả, định nghĩa và hiểu một cách rõ ràng.
- Khahcs hàng có nhu cầu về sản phẩm sớm.

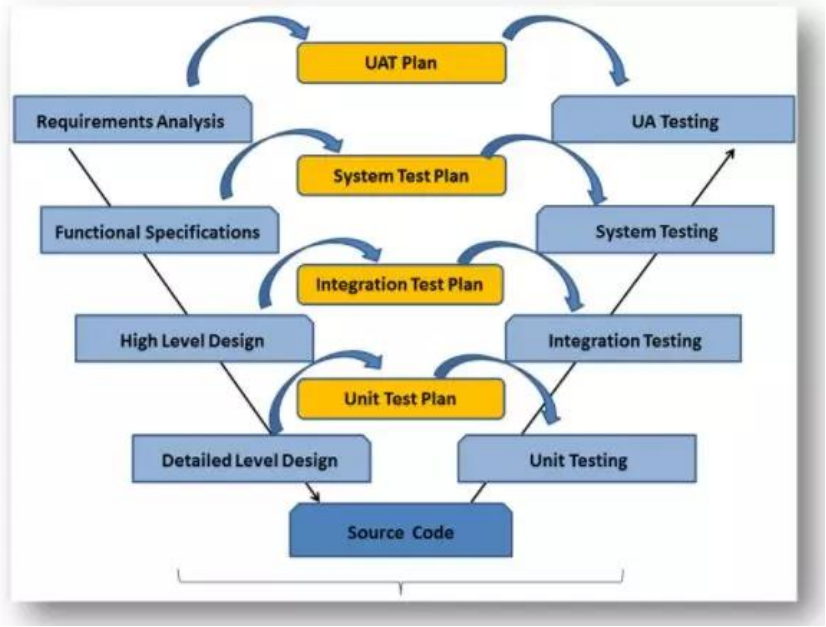
Ưu điểm

- Phát triển nhanh chóng.
- Mô hình này linh hoạt hơn, ít tốn kém hơn khi thay đổi phạm vi và yêu cầu.
- Dễ dàng hơn trong việc kiểm tra và sửa lỗi.

Nhược điểm

- Cần lập plan và thiết kế tốt.
- Tổng chi phí là cao hơn so với mô hình thác nước.

1.6 Mô hình chữ V(V model)



Mô tả

- Mô hình chữ V là một phần mở rộng của mô hình thác nước và được dựa trên sự kết hợp của một giai đoạn thử nghiệm cho từng giai đoạn phát triển tương ứng. Đây là một mô

hình có tính kỷ luật cao và giai đoạn tiếp theo chỉ bắt đầu sau khi hoàn thành giai đoạn trước.

- Với V model thì công việc test được tham gia ngay từ đầu.

Ứng dụng

- Yêu cầu được xác định rõ ràng.
- Xác định sản phẩm ổn định.
- Công nghệ không thay đổi và được hiểu rõ bởi nhóm dự án.
- Không có yêu cầu không rõ ràng hoặc không xác định.
- Dự án ngắn.

Ưu điểm

- Đây là một mô hình có tính kỷ luật cao và các giai đoạn được hoàn thành cùng một lúc.
- Hoạt động tốt cho các dự án nhỏ, khi các yêu cầu được hiểu rất rõ.
- Đơn giản và dễ hiểu và dễ sử dụng, dễ quản lý.

Nhược điểm

- Khó quản lý kiểm soát rủi ro, rủi ro cao.
- Không phải là một mô hình tốt cho các dự án phức tạp và hướng đối tượng.
- Mô hình kém cho các dự án dài và đang diễn ra.
- Không thích hợp cho các dự án có nguy cơ thay đổi yêu cầu trung bình đến cao.

1.7 Mô hình Scrum

Mô tả

- Chia các yêu cầu ra làm theo từng giai đoạn. Mỗi 1 giai đoạn(sprint) chỉ làm 1 số lượng yêu cầu nhất định.

- Mỗi một sprint thường kéo dài từ 1 tuần đến 4 tuần (ko dài hơn 1 tháng).
- Đầu sprint sẽ lên kế hoạch làm những yêu cầu nào. Sau đó, sẽ thực hiện code và test. Cuối sprint là 1 sản phẩm hoàn thiện cả code lẫn test có thể demo và chạy được.
- Hoàn thành sprint 1, tiếp tục làm sprint 2, sprint... cho đến khi hoàn thành hết các yêu cầu.
- Trong mỗi 1 sprint thì sẽ có họp hàng ngày – daily meeting từ 15 – 20 phút. Mỗi thành viên sẽ báo cáo: Hôm qua tôi đã làm gì? Hôm nay tôi sẽ làm gì? Có gặp khó khăn gì không?
- Scrum là mô hình hướng khách hàng (Customer oriented).

Các nhân tố tạo nên quy trình Scrum

Có 3 thành tố quan trọng cấu thành nên SCRUM:

- **Tổ chức (Organization)**
    - Tổ chức nhóm dự án và Roles: Vài trò.
    - Product Owner: Người sở hữu sản phẩm.
    - ScrumMaster: Người điều phối.
    - Development Team: Nhóm phát triển.
- **Tài liệu (Atifacts):** đó chính là các kết quả đầu ra.
    - Product Backlog: Danh sách các chức năng cần phát triển của sản phẩm.
    - Sprint Backlog: Danh sách các chức năng cần phát triển cho mỗi giai đoạn.
    - Estimation:Kết quả ước lượng của team.
- **Qui trình(Process)**: Qui định cách thức vận hành của SCRUM.
    - Sprint Planning meeting: Hoạch định cho mỗi giai đoạn.
    - Review: Tổng kết cho mỗi giai đoạn.
    - Daily Scrum Meeting: Review hàng ngày.

Tổ chức dự án

- **Product Owner**
    - Product Owner là người sở hữu sản phẩm, người quyết định sản phẩm có những chức năng nào và là người quyết định Product Backlog.

    - Thông thường Role này được khách hàng hoặc người đại diện cho khách hàng đảm nhận.

- **ScrumMaster**
    - Scrum Master là người đảm bảo các qui trình của Scrum được thực hiện đúng và thuận lợi.
- **Development Team**
    - Một nhóm từ 4-7 kỹ sư phần mềm chịu trách nhiệm phát triển sản phẩm.
    - Nhóm dự án phải làm việc với Product Owner để quyết định những gì sẽ làm trong Sprint (giai đoạn)này và kết quả sẽ ra sao.
    - Thảo luận để đưa ra các giải pháp, ước lượng thời gian thực hiện công việc, họp đánh giá kết quả công việc.
- **Product Backlog**
    - Product Backlog là danh sách các chức năng cần được phát triển của sản phẩm.
    - Danh sách này do Product Owner quyết định.
    - Thường xuyên được cập nhật để đáp ứng được nhu cầu thay đổi của khách hàng và dự án.

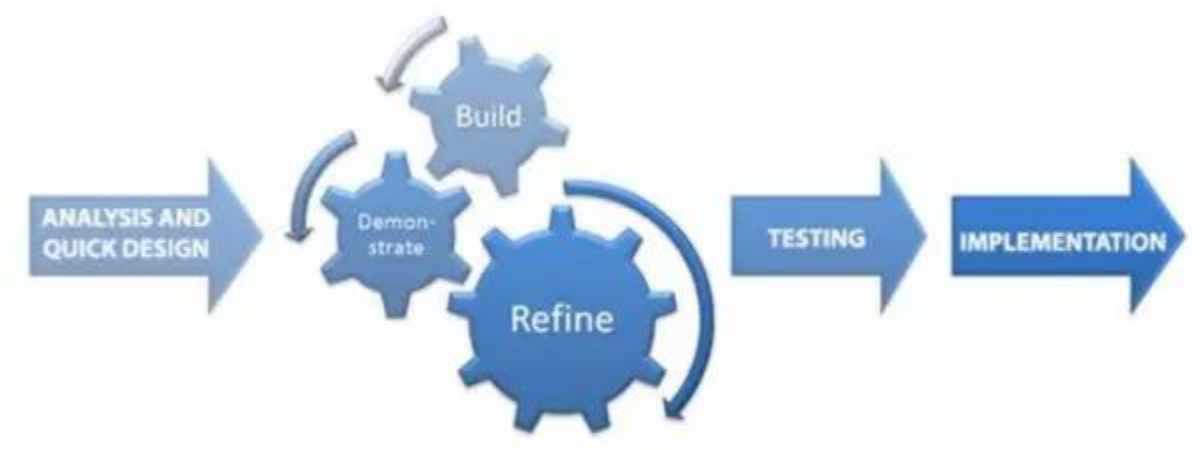
Ưu điểm

- Một người có thể thực hiện nhiều việc ví dụ như dev có thể test.
- Phát hiện lỗi sớm.
- Có khả năng áp dụng được cho những dự án mà yêu cầu khách hàng không rõ ràng ngay từ đầu.

Nhược điểm

- Trình độ của nhóm cần có một kỹ năng nhất định.
- Phải có sự hiểu biết về mô hình aglie.
- Khó khăn trong việc xác định ngân sách và thời gian.
- Luôn nghe ý kiến phản hồi từ khách hàng và thay đổi theo nên thời gian sẽ kéo dài.
- Vai trò của PO rất quan trọng, PO là người định hướng sản phẩm. Nếu PO làm không tốt sẽ ảnh hưởng đến kết quả chung.

1.8 Mô hình RAD



Mô tả

- Mô hình RAD là một phương pháp phát triển phần mềm sử dụng quy hoạch tối thiểu có lợi cho việc tạo mẫu nhanh.
- Các mô-đun chức năng được phát triển song song như nguyên mẫu và được tích hợp để tạo ra sản phẩm hoàn chỉnh để phân phối sản phẩm nhanh hơn.
- Đảm bảo rằng các nguyên mẫu được phát triển có thể tái sử dụng được.

Ứng dụng

Mô hình RAD có thể được áp dụng thành công cho các dự án:

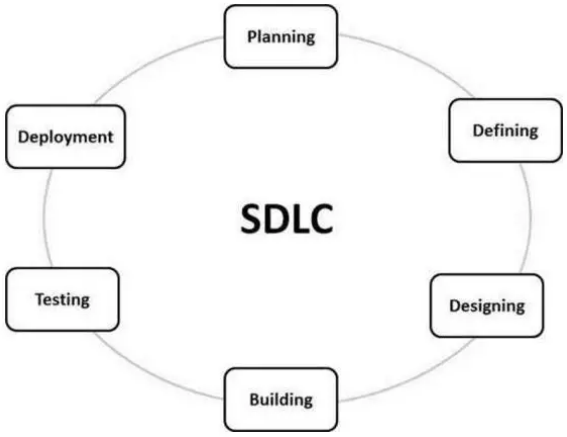
- Module hóa rõ ràng. Nếu dự án không thể được chia thành các mô-đun, RAD có thể không thành công.
- RAD nên được sử dụng khi có nhu cầu để tạo ra một hệ thống có yêu cầu khách hàng thay đổi trong khoảng thời gian nhỏ 2-3 tháng.
- Nên được sử dụng khi đã có sẵn designer cho model và chi phí cao.

Ưu điểm

- Giảm thời gian phát triển.
- Tăng khả năng tái sử dụng của các thành phần.
- Đưa ra đánh giá ban đầu nhanh chóng.
- Khuyến khích khách hàng đưa ra phản hồi.

Nhược điểm

- Trình độ của nhóm cần có một kỹ năng nhất định.
- Chỉ những hệ thống có module mới sử dụng được mô hình này.



Quy trình quản lý dự án Agile-Scrum

1. Khởi đầu dự án (Project Initiation)

- **Hình thành nhóm Scrum**: Lựa chọn Product Owner (PO), Scrum Master, và Development Team.
- **Xây dựng Product Backlog**: PO tạo danh sách các User Stories hoặc tính năng cần phát triển.
- **Xác định mục tiêu dự án**: Thống nhất về mục tiêu và các tiêu chí thành công.

2. Lập kế hoạch Sprint (Sprint Planning)

- **Chọn các mục tiêu Sprint**: PO và team thảo luận để chọn các mục trong Product Backlog cần hoàn thành trong Sprint.
- **Ước tính và chia nhỏ công việc**: Chia các mục thành các task nhỏ hơn để dễ quản lý.
- **Đặt mục tiêu Sprint**: Xác định kết quả cần đạt sau Sprint.

3. Thực hiện Sprint

- **Daily Stand-up**: Cuộc họp hàng ngày (~15 phút) để nhóm cập nhật tiến độ, thảo luận vấn đề và điều chỉnh kế hoạch.
- **Phát triển và kiểm thử**: Team thực hiện các task, hoàn thiện tính năng kèm theo kiểm thử liên tục.
- **Theo dõi tiến độ**: Sử dụng công cụ như Burndown Chart để theo dõi lượng công việc còn lại.

4. Kết thúc Sprint

- **Sprint Review**: Team trình bày các tính năng hoàn thành cho PO và các bên liên quan. Nhận phản hồi và cập nhật Product Backlog.
- **Sprint Retrospective**: Đánh giá hiệu quả công việc, xác định điểm mạnh, hạn chế và cải thiện quy trình.

5. Lặp lại quy trình

- Sau khi kết thúc Sprint, nhóm sẽ tiếp tục Sprint tiếp theo với Product Backlog được cập nhật.

Phân tích một Sprint

1. Thời gian Sprint

- Sprint thường kéo dài từ 1-4 tuần, tùy thuộc vào tính chất dự án và yêu cầu cụ thể.

2. Các hoạt động trong Sprint

- **Lập kế hoạch (Planning)**: Xác định lượng công việc phù hợp với năng lực nhóm.
- **Thực hiện công việc (Execution)**: Hoàn thiện các task với sự hỗ trợ của Scrum Master.
- **Theo dõi và điều chỉnh (Monitoring)**: Scrum Master đảm bảo nhóm không bị cản trở và điều chỉnh công việc nếu cần.
- **Kết thúc (Closure)**: Xác nhận các mục tiêu Sprint đã đạt và chuẩn bị cho Sprint tiếp theo.

3. Kết quả Sprint

- **Increment (Gia tăng giá trị)**: Sản phẩm hoặc tính năng có thể bàn giao được (shippable).
- **Phản hồi từ khách hàng hoặc bên liên quan**: Làm cơ sở để cải tiến liên tục.

4. Đánh giá Sprint

- **Hiệu quả**: Có hoàn thành được tất cả mục tiêu Sprint không?
- **Khả năng dự đoán**: Velocity của team có ổn định không?
- **Chất lượng**: Tính năng có đạt tiêu chuẩn chất lượng không?

WBS là gì và quy trình xây dựng WBS

Quy trình xây dựng WBS

WBS (Work Breakdown Structure) là Cấu trúc phân rã công việc, một công cụ trong quản lý dự án được sử dụng để chia nhỏ dự án thành các phần công việc nhỏ hơn, dễ quản lý hơn. WBS là nền tảng để xác định phạm vi dự án, lập kế hoạch, phân bổ nguồn lực và theo dõi tiến độ.

Đặc điểm của WBS:

- Là một cây phân cấp (hierarchical tree) hiển thị toàn bộ phạm vi dự án.

- Mỗi nhánh của WBS thể hiện một công việc cụ thể, gọi là Work Package.
- Tập trung vào kết quả hoặc sản phẩm đầu ra, không phải hoạt động.
- Mỗi nhánh có thể chia thành nhiều cấp, từ cấp cao (dự án tổng thể) đến cấp chi tiết (nhiệm vụ cụ thể).

Lợi ích của WBS:

1. Quản lý phạm vi: Xác định tất cả các công việc cần thiết để hoàn thành dự án.
2. Phân bổ nguồn lực hiệu quả: Giúp định rõ ai chịu trách nhiệm và cần làm gì.
3. Quản lý tiến độ và chi phí: Hỗ trợ lập kế hoạch và theo dõi.
4. Tránh thiếu sót: Đảm bảo không bỏ sót bất kỳ công việc nào.
5. Giao tiếp: Cung cấp hình ảnh trực quan về dự án cho các bên liên quan.

2. Quy trình xây dựng WBS

Để xây dựng WBS, bạn cần tuân theo quy trình từng bước như sau:

Bước 1: Xác định phạm vi dự án

- Hiểu rõ mục tiêu của dự án và các sản phẩm đầu ra cần đạt được.
- Thu thập yêu cầu dự án từ các bên liên quan.
- Xác định các kết quả đầu ra chính (deliverables) của dự án.
- Đảm bảo rằng phạm vi đã được phê duyệt trong tài liệu khởi tạo dự án (Project Charter).
- Tài liệu tham khảo:
    - Project Charter: Tài liệu khởi động dự án.
    - SOW (Statement of Work): Mô tả công việc.
    - Requirement Documents: Yêu cầu từ khách hàng.

Ví dụ: Dự án phát triển website thương mại điện tử. Mục tiêu: Xây dựng một website với tính năng giỏ hàng, thanh toán trực tuyến, và hệ thống quản lý khách hàng.

Bước 2: Xác định các Deliverables (Sản phẩm đầu ra chính) Phân chia dự án thành các gói công việc lớn

- Xác định các thành phần lớn (Level 1) của dự án, thường là các giai đoạn hoặc mục tiêu chính.
- Chia dự án thành các thành phần lớn theo logic tự nhiên hoặc các giai đoạn.

Ví dụ:

- Thiết kế giao diện người dùng (UI/UX Design).
- Phát triển tính năng chính.
- Kiểm thử và đảm bảo chất lượng.
- Triển khai và đào tạo.

Bước 3: Phân rã Deliverables thành các Work Packages

- Chia nhỏ từng Deliverable thành các công việc cụ thể hơn (Level 2, Level 3, ...).
- Mỗi công việc phải đủ nhỏ để:
    - Có thể hoàn thành trong thời gian hợp lý.
    - Có thể ước lượng được chi phí và thời gian.
- Phân chia từng gói công việc lớn (work packages) thành các công việc nhỏ hơn.
- Tiếp tục phân rã cho đến khi đạt được các công việc mà:
    - Có thể giao cho một cá nhân hoặc nhóm.
    - Dễ dàng ước tính thời gian và chi phí.
    - Có thể đo lường và kiểm soát.

Ví dụ:

- Deliverable: Thiết kế giao diện người dùng
    - Thu thập yêu cầu từ khách hàng.
    - Tạo mockups và wireframes.
    - Phản hồi và chỉnh sửa.
    - Hoàn thiện thiết kế.

Bước 4: Kiểm tra tính đầy đủ của WBS

- Đảm bảo mọi công việc trong phạm vi dự án đã được liệt kê.
- Sử dụng nguyên tắc 100% Rule:
    - WBS phải bao gồm 100% phạm vi công việc và không bỏ sót.
- Kiểm tra tính logic, liên kết giữa các cấp của WBS.
- Gắn mã số hoặc mã nhận diện cho từng thành phần của WBS để quản lý dễ dàng hơn.

- Ví dụ:
    - 1.0: Giai đoạn khởi tạo.
    - 1.1: Phân tích yêu cầu.
    - 1.1.1: Phỏng vấn khách hàng.

Ví dụ: Trong dự án xây dựng website, nếu thiếu phần "Bảo trì sau triển khai", WBS sẽ không đầy đủ.

Bước 5: Mã hóa công việc (Coding WBS Elements)

- Gắn mã nhận diện cho từng Work Package để dễ quản lý và theo dõi.
- Sử dụng mã số hoặc ký hiệu (ví dụ: 1.1, 1.2, 2.1).
- Kiểm tra lại để đảm bảo rằng tất cả các công việc cần thiết đã được bao gồm.
- Đảm bảo WBS tuân theo các nguyên tắc:
    - Không trùng lặp nhiệm vụ.
    - Đảm bảo tất cả các công việc là kết quả định hướng.
- Trình bày cho các bên liên quan để phê duyệt.

Ví dụ:

1. Thiết kế giao diện người dùng
1. 1.1 Thu thập yêu cầu.
2. 1.2 Tạo mockups.
3. 1.3 Chỉnh sửa.

Bước 6: Đưa vào công cụ quản lý dự án

- Nhập WBS vào các công cụ hỗ trợ quản lý dự án như:
    - Microsoft Project.
    - Trello.
    - Jira.
    - Asana.
- Sử dụng WBS làm cơ sở để:
    - Xây dựng lịch trình (Timeline).

- Phân bổ nguồn lực (Resources Allocation).
- Dự đoán chi phí (Cost Estimation).
- Theo dõi tiến độ và kiểm soát rủi ro.

Ví dụ minh họa WBS

Ví dụ WBS cho dự án phát triển phần mềm

Cấp độ 1: Dự án phát triển phần mềm

- 1.0 Thu thập yêu cầu
    - 1.1 Phỏng vấn khách hàng.
    - 1.2 Phân tích yêu cầu.
    - 1.3 Tạo tài liệu yêu cầu (SRS).
- 2.0 Thiết kế hệ thống
    - 2.1 Thiết kế cơ sở dữ liệu.
    - 2.2 Thiết kế giao diện người dùng.
    - 2.3 Xác minh và phê duyệt thiết kế.
- 3.0 Phát triển phần mềm
    - 3.1 Lập trình chức năng chính.
    - 3.2 Tích hợp module.
    - 3.3 Kiểm tra đơn vị (Unit Testing).
- 4.0 Kiểm thử
    - 4.1 Kiểm thử hệ thống.
    - 4.2 Kiểm thử chấp nhận người dùng (UAT).
- 5.0 Triển khai
    - 5.1 Cài đặt phần mềm trên môi trường sản xuất.
    - 5.2 Đào tạo người dùng.

Lợi ích của WBS

1. Giúp quản lý phạm vi dự án hiệu quả: Đảm bảo rằng tất cả các công việc cần thiết được xác định.
2. Tăng tính minh bạch: Làm rõ trách nhiệm của từng cá nhân/nhóm.
3. Hỗ trợ lập kế hoạch: Làm nền tảng cho việc ước tính thời gian, chi phí, và nguồn lực.
4. Giúp theo dõi tiến độ: Dễ dàng so sánh thực tế với kế hoạch ban đầu.
5. Giảm rủi ro: Tránh bỏ sót công việc và giảm nguy cơ phát sinh chi phí hoặc thời gian không cần thiết.

Lưu ý quan trọng khi xây dựng WBS:

1. Không phân rã quá mức: Tránh chia nhỏ công việc đến mức không cần thiết.
2. Tập trung vào kết quả: Mỗi nhánh của WBS phải đại diện cho sản phẩm đầu ra.

Đồng bộ với các bên liên quan: Luôn kiểm tra lại với khách hàng hoặc quản lý cấp trên để đảm bảo sự đồng thuận.

Ví dụ WBS cho dự án phát triển phần mềm

Cấp độ 1: Dự án phát triển phần mềm

- **1.0 Thu thập yêu cầu**
    - 1.1 Phỏng vấn khách hàng.
    - 1.2 Phân tích yêu cầu.
    - 1.3 Tạo tài liệu yêu cầu (SRS).
- **2.0 Thiết kế hệ thống**
    - 2.1 Thiết kế cơ sở dữ liệu.
    - 2.2 Thiết kế giao diện người dùng.
    - 2.3 Xác minh và phê duyệt thiết kế.
- **3.0 Phát triển phần mềm**
    - 3.1 Lập trình chức năng chính.
    - 3.2 Tích hợp module.
    - 3.3 Kiểm tra đơn vị (Unit Testing).
- **4.0 Kiểm thử**
    - 4.1 Kiểm thử hệ thống.

    - 4.2 Kiểm thử chấp nhận người dùng (UAT).
- **5.0 Triển khai**
    - 5.1 Cài đặt phần mềm trên môi trường sản xuất.
    - 5.2 Đào tạo người dùng.

Lợi ích của WBS

1. **Giúp quản lý phạm vi dự án hiệu quả**: Đảm bảo rằng tất cả các công việc cần thiết được xác định.
2. **Tăng tính minh bạch**: Làm rõ trách nhiệm của từng cá nhân/nhóm.
3. **Hỗ trợ lập kế hoạch**: Làm nền tảng cho việc ước tính thời gian, chi phí, và nguồn lực.
4. **Giúp theo dõi tiến độ**: Dễ dàng so sánh thực tế với kế hoạch ban đầu.
5. **Giảm rủi ro**: Tránh bỏ sót công việc và giảm nguy cơ phát sinh chi phí hoặc thời gian không cần thiết.

Các kỹ thuật ước lượng thời gian, ưu điểm, hạn chế, ví dụ.

Trong quản lý dự án, ước lượng thời gian là một bước quan trọng để lập kế hoạch, phân bổ nguồn lực và quản lý tiến độ. Dưới đây là các kỹ thuật ước lượng thời gian, kèm theo ưu điểm, hạn chế và ví dụ cụ thể.

---

1. Ước lượng tương tự (Analogous Estimation)

Mô tả:

Phương pháp này dựa trên việc so sánh với các dự án tương tự đã hoàn thành trước đó. Các thông tin lịch sử, như thời gian hoàn thành, nguồn lực sử dụng và mức độ phức tạp, được sử dụng để suy ra thời gian cho dự án hiện tại. Điều này đòi hỏi phải có một cơ sở dữ liệu về các dự án trước để làm mẫu so sánh.

Ưu điểm:

Ưu điểm lớn của kỹ thuật này là tính thực tiễn và dễ thực hiện. Dựa trên dữ liệu từ các dự án trước, nhóm dự án có thể nhanh chóng đưa ra ước lượng mà không cần quá nhiều phân tích chi tiết. Phương pháp này cũng có khả năng tạo ra các dự đoán sát với thực tế nếu dự án mới có nhiều điểm tương đồng với các dự án cũ.

Hạn chế:

Hạn chế của phương pháp này nằm ở tính tương đồng. Nếu dự án mới không có đủ điểm giống với dự án cũ, kết quả ước lượng có thể không chính xác. Ngoài ra, nếu dữ liệu lịch sử

không đầy đủ hoặc không chính xác, quá trình so sánh cũng sẽ bị ảnh hưởng, dẫn đến sai lệch.

Ví dụ:

Một công ty phát triển phần mềm từng xây dựng một ứng dụng thương mại điện tử cơ bản mất 6 tháng. Khi nhận một dự án mới với yêu cầu tương tự, họ sử dụng thông tin từ dự án trước đó để ước lượng thời gian cần thiết cho dự án mới cũng sẽ mất khoảng 6 tháng.

2. Ước lượng tham số (Parametric Estimation)

Mô tả:

Phương pháp phân tích tham số sử dụng các công thức toán học hoặc các tham số đo lường cụ thể để ước lượng thời gian. Dựa trên dữ liệu thống kê và các tiêu chuẩn ngành, các nhà quản lý có thể tính toán thời gian bằng cách nhân số lượng công việc với năng suất dự kiến. Ví dụ: thời gian cần để viết một dòng mã (LOC) hoặc thời gian trung bình hoàn thiện một module.

Ưu điểm:

Ưu điểm của phương pháp này là tính chính xác cao và khả năng chuẩn hóa. Vì dựa trên dữ liệu đo lường thực tế, kỹ thuật này có thể cung cấp ước lượng sát với thực tế nếu tham số đầu vào chính xác. Phương pháp này cũng cho phép dễ dàng áp dụng và so sánh giữa các dự án khác nhau.

Hạn chế:

Tuy nhiên, phương pháp này phụ thuộc nhiều vào chất lượng dữ liệu đầu vào. Nếu dữ liệu không chính xác hoặc không phù hợp với dự án hiện tại, kết quả sẽ bị sai lệch. Ngoài ra, đối với các dự án mới hoặc chưa có dữ liệu tham số, phương pháp này trở nên khó áp dụng.

Ví dụ:

Nếu một nhóm phát triển ước tính mất 2 giờ để viết và kiểm thử 100 dòng mã, thì với một dự án có 10.000 dòng mã, họ có thể ước lượng rằng sẽ mất khoảng 200 giờ.

---

3. Ước lượng ba điểm (Three-Point Estimation)

Mô tả:
Dựa trên ba giá trị ước lượng:

- **Tối ưu (Optimistic - O)**: Thời gian ít nhất cần thiết.
- **Bi quan (Pessimistic - P)**: Thời gian nhiều nhất cần thiết.
- **Khả thi (Most Likely - M)**: Thời gian có khả năng xảy ra nhất.

Công thức:

$$\text{Ước lượng} = \frac{O + 4M + P}{6}$$

Ưu điểm:

- Tính đến cả rủi ro và tính không chắc chắn.
- Chính xác hơn so với các phương pháp đơn giản.

Hạn chế:

- Yêu cầu nhiều dữ liệu và đánh giá chủ quan.
- Tốn thời gian hơn để tính toán.

Ví dụ:

- O = 5 ngày, M = 7 ngày, P = 10 ngày.
- Thời gian ước lượng: $\frac{5+4(7)+10}{6} = 7.17$ ngày.

---

4. Kỹ thuật PERT (Program Evaluation and Review Technique)

Mô tả:
Là một dạng mở rộng của ước lượng ba điểm, tập trung vào việc ước lượng thời gian cho từng công việc trong sơ đồ mạng và tính tổng thời gian dự án.

Công thức:

$$\text{Thời gian dự kiến} = \frac{O + 4M + P}{6}$$

Ưu điểm:

- Phù hợp cho các dự án phức tạp với nhiều nhiệm vụ liên quan.
- Tính đến tính không chắc chắn trong từng công việc.

Hạn chế:

- Cần phân tích chi tiết từng công việc, mất thời gian.
- Phức tạp hơn các kỹ thuật khác.

Ví dụ:
Trong sơ đồ mạng, các nhiệm vụ A, B, C có thời gian dự kiến và mối quan hệ phụ thuộc, tổng thời gian dự án được tính từ đường dẫn quan trọng (Critical Path).

---

5. Kỹ thuật ước lượng theo nhóm (Group Decision-Based Estimation)

Mô tả:
Sử dụng kinh nghiệm của một nhóm chuyên gia để đưa ra ước lượng.

Ưu điểm:

- Tận dụng được nhiều kinh nghiệm và góc nhìn khác nhau.
- Độ tin cậy cao nếu nhóm có chuyên môn tốt.

Hạn chế:

- Phụ thuộc vào sự phối hợp và chất lượng của nhóm.
- Có thể bị ảnh hưởng bởi ý kiến chủ quan hoặc xung đột trong nhóm.

Ví dụ:
Một nhóm chuyên gia ước tính rằng nhiệm vụ phát triển ứng dụng sẽ mất từ 4 đến 6 tuần dựa trên kinh nghiệm của họ.

---

6. Ước lượng từ dưới lên (Bottom-Up Estimation)

Mô tả:
Ước lượng thời gian cho từng nhiệm vụ nhỏ, sau đó cộng dồn để có tổng thời gian dự án.

Ưu điểm:

- Chính xác vì tập trung vào từng chi tiết nhỏ.
- Giúp nhận diện các công việc cụ thể.

Hạn chế:

- Tốn nhiều thời gian và công sức.
- Khó thực hiện nếu dự án lớn hoặc không có đủ thông tin.

Ví dụ:
Nếu 3 nhiệm vụ cần lần lượt 2 giờ, 4 giờ, và 6 giờ, tổng thời gian là 2+4+6=12

---

7. Ước lượng Delphi (Delphi Technique)

Mô tả:
Một nhóm chuyên gia độc lập đưa ra ước lượng, sau đó thảo luận để đạt được sự đồng thuận.

Ưu điểm:

- Loại bỏ sự thiên vị trong nhóm.
- Có tính chính xác cao nếu các chuyên gia có kinh nghiệm tốt.

Hạn chế:

- Tốn thời gian cho việc thảo luận và điều chỉnh.
- Phụ thuộc vào chất lượng chuyên gia.

Ví dụ:
Một nhóm chuyên gia ước tính thời gian phát triển phần mềm trong 5 giai đoạn và thống nhất rằng toàn bộ dự án sẽ mất 8 tháng.

So sánh ưu điểm và hạn chế

Kỹ thuật	Ưu điểm	Hạn chế	Ví dụ thích hợp
Ước lượng tương tự	Nhanh, đơn giản	Dễ sai nếu không có dự án tương tự	Dự án có tính chất lặp lại
Ước lượng tham số	Chính xác nếu tham số tốt	Phụ thuộc vào dữ liệu chất lượng	Dự án có dữ liệu chuẩn hóa
Ước lượng ba điểm	Tính đến rủi ro	Phụ thuộc vào đánh giá chủ quan	Dự án có mức độ rủi ro trung bình
PERT	Phù hợp dự án phức tạp	Tốn thời gian, phức tạp	Quản lý dự án lớn, nhiều công việc
Theo nhóm	Kết hợp nhiều kinh nghiệm	Có thể gây tranh luận hoặc chậm trễ	Khi cần ý kiến của nhiều chuyên gia
Từ dưới lên	Chính xác cao	Tốn nhiều thời gian và tài nguyên	Dự án nhỏ hoặc cần chi tiết
Delphi	Loại bỏ thiên vị, đồng thuận cao	Tốn thời gian, phụ thuộc chất lượng chuyên gia	Dự án phức tạp, cần sự thống nhất

Các phương pháp lập lịch biểu về tiến độ thực hiện dự án.

Lập lịch biểu tiến độ thực hiện dự án là một bước quan trọng để đảm bảo các công việc được hoàn thành đúng thời gian, tối ưu hóa nguồn lực và giảm thiểu rủi ro. Dưới đây là các phương pháp lập lịch biểu phổ biến, kèm theo ưu điểm, hạn chế và ví dụ.

1. Biểu đồ Gantt (Gantt Chart)

Mô tả:
Là một dạng biểu đồ thanh biểu diễn các công việc theo thời gian, với trục ngang thể hiện thời gian và trục dọc thể hiện các công việc.

Ưu điểm:

- Trực quan và dễ hiểu.
- Hiển thị rõ ràng các mốc thời gian, thời gian bắt đầu/kết thúc và mối quan hệ giữa các công việc.
- Hỗ trợ theo dõi tiến độ thực hiện.

Hạn chế:

- Khó quản lý nếu dự án lớn và phức tạp.
- Không cung cấp thông tin chi tiết về các đường dẫn quan trọng.

Ví dụ:
Biểu đồ Gantt cho dự án phát triển phần mềm bao gồm các nhiệm vụ như "Phân tích yêu cầu (10 ngày)", "Thiết kế giao diện (15 ngày)", và "Phát triển chức năng (30 ngày)".

---

2. Phương pháp đường găng (Critical Path Method - CPM)

Mô tả:
Là kỹ thuật xác định đường dẫn quan trọng nhất trong dự án, bao gồm các công việc phải hoàn thành đúng hạn để không làm trễ tiến độ chung.

Ưu điểm:

- Xác định các nhiệm vụ quan trọng nhất.
- Giúp quản lý rủi ro bằng cách tập trung vào đường găng.
- Phù hợp với các dự án lớn và phức tạp.

Hạn chế:

- Đòi hỏi phân tích chi tiết các nhiệm vụ và thời gian.
- Không tính đến tính linh hoạt trong lịch trình.

Ví dụ:
Một dự án xây dựng có các công việc: "Thiết kế (5 ngày)", "Đổ móng (10 ngày)", và "Xây dựng khung (15 ngày)". CPM xác định "Đổ móng" và "Xây dựng khung" nằm trên đường găng.

---

3. Phương pháp PERT (Program Evaluation and Review Technique)

Mô tả:
Là kỹ thuật lập lịch dựa trên ước lượng ba điểm (tối ưu, khả thi, bi quan) để tính toán thời gian dự kiến cho từng nhiệm vụ và tổng thời gian dự án.

Ưu điểm:

- Tính đến sự không chắc chắn trong ước lượng thời gian.
- Hữu ích cho các dự án có nhiều yếu tố rủi ro.
- Kết hợp chặt chẽ với CPM để xác định đường găng.

Hạn chế:

- Phức tạp hơn so với CPM.
- Đòi hỏi nhiều dữ liệu đầu vào.

Ví dụ:

Một nhiệm vụ có:

- Thời gian lạc quan (O): 4 ngày,
- Khả thi (M): 6 ngày,
- Bi quan (P): 10 ngày.
  Thời gian dự kiến: $\frac{4+4(6)+10}{6} = 6.67 \text{ ngày}$.

---

4. Phương pháp Sơ đồ mạng (Network Diagram)

Mô tả:
Biểu diễn các công việc và mối quan hệ phụ thuộc giữa chúng thông qua sơ đồ nút và mũi tên (như sơ đồ AOA - Activity on Arrow hoặc AON - Activity on Node).

Ưu điểm:

- Hiển thị mối quan hệ logic giữa các công việc.
- Dễ dàng xác định đường găng và thời gian hoàn thành dự án.

Hạn chế:

- Khó đọc nếu dự án có nhiều nhiệm vụ và phụ thuộc phức tạp.
- Tốn thời gian để xây dựng sơ đồ.

Ví dụ:
Công việc A (3 ngày) -> Công việc B (2 ngày) -> Công việc C (5 ngày). Sơ đồ mạng sẽ biểu diễn các công việc này và sự phụ thuộc giữa chúng.

---

5. Kỹ thuật Agile/Scrum

Mô tả:
Dự án được chia thành các giai đoạn ngắn (Sprint), mỗi Sprint có một lịch trình cụ thể với các nhiệm vụ được ưu tiên.

Ưu điểm:

- Linh hoạt và dễ dàng thích nghi với thay đổi yêu cầu.
- Phù hợp với các dự án phát triển phần mềm hoặc sáng tạo.
- Giúp cải thiện sự cộng tác và minh bạch trong nhóm.

Hạn chế:

- Không phù hợp với các dự án có yêu cầu cố định.
- Đòi hỏi sự phối hợp chặt chẽ giữa các thành viên.

Ví dụ:
Một Sprint 2 tuần có các nhiệm vụ: "Phát triển module A (3 ngày)", "Kiểm thử module A (2 ngày)", và "Phát triển module B (4 ngày)".

---

6. Kỹ thuật Lập lịch tuyến tính (Linear Scheduling Method - LSM)

Mô tả:
Phù hợp với các dự án có nhiệm vụ thực hiện liên tục, như xây dựng đường, cầu, hoặc sản xuất hàng loạt.

Ưu điểm:

- Hiệu quả cho các dự án tuyến tính.
- Hiển thị rõ sự phối hợp giữa các nhiệm vụ lặp lại.

Hạn chế:

- Không phù hợp cho các dự án phức tạp hoặc không tuyến tính.

Ví dụ:
Dự án xây dựng đường có các công việc: "San lấp mặt bằng (10 ngày)", "Đổ nền đường (15 ngày)", và "Lát đường (20 ngày)".

---

7. Phương pháp Lập lịch tài nguyên (Resource Scheduling)

Mô tả:
Lập lịch dựa trên sự sẵn có của tài nguyên, như nhân lực, thiết bị, hoặc nguyên vật liệu.

Ưu điểm:

- Giảm tình trạng quá tải tài nguyên.
- Giúp tối ưu hóa việc sử dụng tài nguyên.

Hạn chế:

- Có thể làm chậm tiến độ nếu tài nguyên hạn chế.
- Cần phối hợp chặt chẽ với các bên liên quan.

Ví dụ:
Một kỹ sư chỉ có thể làm việc tối đa 8 giờ/ngày. Lập lịch cần đảm bảo không vượt quá giới hạn này.

So sánh các phương pháp lập lịch biểu

Phương pháp	Ưu điểm	Hạn chế	Ứng dụng phù hợp
Biểu đồ Gantt	Trực quan, dễ sử dụng	Khó quản lý dự án lớn	Dự án nhỏ hoặc trung bình
CPM	Xác định đường găng, quản lý rủi ro tốt	Không linh hoạt	Dự án có nhiều nhiệm vụ phụ thuộc logic
PERT	Tính đến rủi ro, không chắc chắn	Phức tạp hơn CPM	Dự án có tính không chắc chắn cao
Sơ đồ mạng	Hiển thị logic giữa các công việc	Khó đọc với dự án phức tạp	Dự án cần phân tích mối quan hệ phụ thuộc
Agile/Scrum	Linh hoạt, thích nghi tốt với thay đổi	Đòi hỏi sự phối hợp chặt chẽ	Phát triển phần mềm, sáng tạo
Lập lịch tuyến tính (LSM)	Phù hợp với dự án lặp lại hoặc liên tục	Không áp dụng cho dự án phức tạp	Dự án xây dựng, sản xuất
Lập lịch tài nguyên	Tối ưu hóa tài nguyên	Làm chậm tiến độ nếu tài nguyên hạn chế	Dự án bị giới hạn tài nguyên

Các phương pháp ước lượng chi phí ngân sách cho dự án

Ước lượng chi phí ngân sách là một phần thiết yếu trong quản lý dự án, giúp xác định các nguồn lực tài chính cần thiết để hoàn thành dự án. Dưới đây là các phương pháp phổ biến được sử dụng để ước lượng chi phí ngân sách, cùng với ưu điểm, hạn chế, và ví dụ minh họa.

---

1. Ước lượng từ dưới lên (Bottom-Up Estimation)

Mô tả:
Phân chia dự án thành các công việc nhỏ hơn, ước lượng chi phí cho từng công việc, sau đó cộng dồn để có tổng ngân sách.

Ưu điểm:

- Chính xác, vì tập trung vào các chi tiết nhỏ.
- Hiểu rõ hơn về các yếu tố chi phí trong dự án.

Hạn chế:

- Mất thời gian và công sức để phân tích từng công việc.
- Dễ bỏ sót chi phí nếu không phân chia kỹ lưỡng.

Ví dụ:
Một dự án xây dựng có các hạng mục như vật liệu (100 triệu), nhân công (50 triệu), thiết bị (20 triệu). Tổng chi phí dự án là 170 triệu.

---

2. Ước lượng tương tự (Analogous Estimation)

Mô tả:
Dựa trên dữ liệu từ các dự án tương tự trước đây để ước lượng chi phí cho dự án hiện tại.

Ưu điểm:

- Nhanh, dễ thực hiện.
- Tận dụng dữ liệu lịch sử để đưa ra dự đoán.

Hạn chế:

- Không chính xác nếu dự án hiện tại khác biệt với dự án tham chiếu.
- Phụ thuộc vào chất lượng và độ tin cậy của dữ liệu quá khứ.

Ví dụ:
Nếu dự án xây dựng trước đây có chi phí 500 triệu cho một khu nhà, một dự án tương tự có thể được ước lượng chi phí là 500 triệu.

---

3. Ước lượng tham số (Parametric Estimation)

Mô tả:
Sử dụng các tham số và công thức để ước lượng chi phí dựa trên các yếu tố liên quan. Ví dụ: chi phí theo mét vuông, số giờ lao động, hoặc số lượng sản phẩm.

Ưu điểm:

- Phù hợp cho các dự án có quy mô lớn.
- Chính xác nếu các tham số được chọn đúng.

Hạn chế:

- Phụ thuộc vào dữ liệu chất lượng cao và mô hình phù hợp.
- Không hiệu quả cho các dự án phức tạp hoặc sáng tạo.

Ví dụ:
Nếu chi phí trung bình để xây dựng $1m^2$ là 5 triệu, chi phí xây dựng một căn nhà $100m^2$ sẽ là $100 \times 5 = 500$ triệu

---

4. Ước lượng ba điểm (Three-Point Estimation)

Mô tả:
Kết hợp ba giá trị ước lượng để tính chi phí:

- **Tối ưu (Optimistic - O)**: Chi phí thấp nhất.
- **Khả thi (Most Likely - M)**: Chi phí khả năng xảy ra cao nhất.
- **Bi quan (Pessimistic - P)**: Chi phí cao nhất.

Công thức:

$$\text{Ước lượng chi phí} = \frac{O + 4M + P}{6}$$

Ưu điểm:

- Tính đến rủi ro và sự không chắc chắn trong ước lượng.
- Phù hợp với các dự án có mức độ biến động cao.

Hạn chế:

- Đòi hỏi nhiều dữ liệu đầu vào.
- Phụ thuộc vào đánh giá chủ quan.

Ví dụ:
Một công việc có chi phí: O = 50 triệu, M = 70 triệu, P = 100 triệu.
Chi phí ước lượng: $\frac{50+4(70)+100}{6} = 73.33 \text{ triệu}.$

---

5. Phân tích giá trị hiện tại (Earned Value Analysis - EVA)

Mô tả:
Dựa trên giá trị thực hiện công việc để so sánh chi phí thực tế với chi phí dự kiến, từ đó ước lượng chi phí còn lại.

Ưu điểm:

- Hiệu quả trong việc kiểm soát ngân sách trong quá trình thực hiện.
- Dễ dàng theo dõi và điều chỉnh chi phí.

Hạn chế:

- Yêu cầu theo dõi sát sao các số liệu chi phí.
- Phức tạp đối với các dự án lớn.

Ví dụ:
Dự án có ngân sách dự kiến là 1 tỷ, nhưng sau 50% tiến độ, đã tiêu hết 600 triệu. EVA giúp tính toán liệu dự án có thể hoàn thành trong ngân sách hay không.

---

6. Ước lượng theo nhóm (Group Decision-Based Estimation)

Mô tả:
Một nhóm chuyên gia đưa ra các dự đoán về chi phí, sau đó thảo luận để đạt được sự đồng thuận.

Ưu điểm:

- Tận dụng nhiều kinh nghiệm và quan điểm khác nhau.
- Giảm thiểu sai sót từ một cá nhân.

Hạn chế:

- Phụ thuộc vào chất lượng của nhóm chuyên gia.
- Có thể mất thời gian để đạt được sự đồng thuận.

Ví dụ:
Nhóm chuyên gia ước tính rằng dự án phần mềm sẽ mất từ 300 triệu đến 500 triệu, và thống nhất chi phí là 400 triệu.

---

7. Ước lượng dự phòng (Contingency Estimation)

Mô tả:
Thêm một khoản chi phí dự phòng vào ngân sách để đối phó với các rủi ro không lường trước.

Ưu điểm:

- Tăng khả năng hoàn thành dự án đúng ngân sách.
- Hữu ích trong các dự án có mức độ không chắc chắn cao.

Hạn chế:

- Có thể dẫn đến lãng phí nếu không xảy ra rủi ro.
- Đòi hỏi phân tích rủi ro chi tiết.

Ví dụ:
Nếu ngân sách dự kiến là 1 tỷ, thêm 10% dự phòng, tổng ngân sách sẽ là 1.1 tỷ.

---

8. Ước lượng theo dữ liệu lịch sử (Historical Data Estimation)

Mô tả:
Sử dụng dữ liệu từ các dự án tương tự trong quá khứ để ước lượng chi phí.

Ưu điểm:

- Nhanh chóng và hiệu quả.
- Tận dụng dữ liệu sẵn có.

Hạn chế:

- Không chính xác nếu dự án hiện tại khác biệt đáng kể.
- Phụ thuộc vào tính đầy đủ và chính xác của dữ liệu quá khứ.

Ví dụ:
Một dự án phần mềm trước đây với quy mô tương tự có chi phí 500 triệu, dự án hiện tại được ước lượng theo chi phí này.

So sánh các phương pháp

Phương pháp	Ưu điểm	Hạn chế	Thích hợp cho
Bottom-Up	Chính xác cao	Mất thời gian, tốn công sức	Dự án lớn, yêu cầu chi tiết
Analogous	Nhanh, dễ thực hiện	Không chính xác với dự án khác biệt	Dự án nhỏ hoặc tương tự quá khứ

Phương pháp	Ưu điểm	Hạn chế	Thích hợp cho
Parametric	Chính xác nếu tham số phù hợp	Đòi hỏi dữ liệu chất lượng cao	Dự án quy mô lớn, định lượng rõ ràng
Three-Point	Tính đến rủi ro	Yêu cầu nhiều dữ liệu	Dự án có sự biến động cao
EVA	Kiểm soát chi phí hiệu quả	Phức tạp với dự án lớn	Dự án trong quá trình thực hiện
Group Estimation	Tận dụng ý kiến chuyên gia	Tốn thời gian, phụ thuộc chất lượng nhóm	Dự án phức tạp, cần kinh nghiệm nhóm
Contingency	Tăng tính an toàn	Có thể lãng phí nếu không xảy ra rủi ro	Dự án rủi ro cao
Historical Data	Dễ dàng, nhanh chóng	Phụ thuộc vào dữ liệu quá khứ	Dự án có lịch sử tương tự

Các nguyên tắc khi sử dụng phương pháp ưu tiên khi phân phối nguồn lực cho dự án

Khi sử dụng phương pháp ưu tiên để phân phối nguồn lực trong dự án, việc tuân thủ các nguyên tắc quan trọng là chìa khóa để đảm bảo rằng nguồn lực được sử dụng hiệu quả, đáp ứng mục tiêu dự án và hạn chế rủi ro. Dưới đây là các nguyên tắc cơ bản khi áp dụng phương pháp ưu tiên:

---

1. Xác định rõ các mục tiêu và yêu cầu của dự án

- **Mô tả**: Các mục tiêu chính của dự án và yêu cầu cụ thể phải được xác định rõ ràng để hướng dẫn việc phân bổ nguồn lực.
- **Áp dụng**: Đảm bảo mọi thành viên trong nhóm hiểu rõ thứ tự ưu tiên của các nhiệm vụ và tầm quan trọng của chúng.
- **Lợi ích**: Giúp tập trung nguồn lực vào các công việc có giá trị cao, góp phần hoàn thành mục tiêu quan trọng nhất.

Ví dụ: Trong một dự án phát triển phần mềm, ưu tiên hoàn thành chức năng chính trước khi phát triển các tính năng bổ sung.

---

2. Đánh giá mức độ quan trọng và tác động của từng công việc

- **Mô tả**: Phân tích và đánh giá mức độ quan trọng (priority) của từng nhiệm vụ dựa trên tác động của chúng đến tiến độ và chất lượng dự án.

- **Áp dụng**: Sử dụng các phương pháp như phân tích giá trị, mô hình RICE (Reach, Impact, Confidence, Effort), hoặc MoSCoW (Must Have, Should Have, Could Have, Won't Have).
- **Lợi ích**: Phân bổ nguồn lực hợp lý, tránh tập trung quá mức vào những công việc ít quan trọng.

Ví dụ: Công việc triển khai hệ thống bảo mật sẽ được ưu tiên cao hơn so với thiết kế giao diện người dùng.

---

3. Xác định các ràng buộc về nguồn lực

- **Mô tả**: Xác định và hiểu rõ các giới hạn về thời gian, nhân sự, tài chính, và công nghệ của dự án.
- **Áp dụng**: Lập danh sách các nguồn lực hiện có và phân tích khả năng đáp ứng cho từng công việc.
- **Lợi ích**: Đảm bảo rằng các nguồn lực không bị quá tải và được sử dụng hiệu quả.

Ví dụ: Nếu chỉ có 2 lập trình viên chính, lịch làm việc sẽ được sắp xếp sao cho họ không phải xử lý quá nhiều nhiệm vụ cùng lúc.

---

4. Xây dựng tiêu chí ưu tiên rõ ràng

- **Mô tả**: Đặt ra tiêu chí cụ thể để xác định mức độ ưu tiên, như thời hạn gấp, tác động lớn đến dự án, hoặc yêu cầu pháp lý bắt buộc.
- **Áp dụng**: Áp dụng ma trận Eisenhower (Gấp/Gấp và Quan trọng/Không quan trọng) hoặc các công cụ tương tự để sắp xếp thứ tự ưu tiên.
- **Lợi ích**: Tránh xung đột trong việc phân bổ nguồn lực và tạo ra sự nhất quán khi ra quyết định.

Ví dụ: Trong dự án xây dựng, công việc liên quan đến đảm bảo an toàn lao động được ưu tiên cao vì yêu cầu pháp lý và rủi ro cao.

---

5. Đảm bảo sự cân bằng trong phân bổ nguồn lực

- **Mô tả**: Nguồn lực cần được phân bổ một cách đồng đều để tránh hiện tượng quá tải ở một số thành viên hoặc đội nhóm, đồng thời hạn chế lãng phí nguồn lực khác.
- **Áp dụng**: Sử dụng các công cụ quản lý dự án như Gantt Chart hoặc Resource Leveling để tối ưu hóa lịch biểu.
- **Lợi ích**: Giảm thiểu nguy cơ kiệt sức của đội ngũ và tối đa hóa năng suất làm việc.

Ví dụ: Trong một dự án xây dựng, đội thi công và đội thiết kế làm việc luân phiên để không xảy ra tình trạng chờ đợi lẫn nhau.

6. Tích hợp các rủi ro và yếu tố không chắc chắn

- **Mô tả**: Lường trước các rủi ro có thể xảy ra và thêm các biện pháp dự phòng khi phân phối nguồn lực.
- **Áp dụng**: Dự trù nguồn lực hoặc thời gian đệm cho các công việc có nguy cơ cao.
- **Lợi ích**: Tăng khả năng ứng phó với các tình huống bất ngờ mà không làm chậm tiến độ dự án.

Ví dụ: Thêm 10% ngân sách dự phòng cho các chi phí phát sinh không lường trước.

---

7. Đánh giá lại và điều chỉnh khi cần thiết

- **Mô tả**: Liên tục giám sát và đánh giá việc phân bổ nguồn lực, từ đó điều chỉnh phù hợp nếu có thay đổi về ưu tiên hoặc các yếu tố khác.
- **Áp dụng**: Sử dụng các cuộc họp định kỳ hoặc các công cụ giám sát để theo dõi tiến độ và mức độ sử dụng nguồn lực.
- **Lợi ích**: Đảm bảo nguồn lực được phân bổ linh hoạt và dự án không bị đình trệ.

Ví dụ: Nếu một thành viên gặp vấn đề sức khỏe và không thể hoàn thành nhiệm vụ, nhóm quản lý sẽ phân bổ lại nhiệm vụ cho người khác.

---

8. Sử dụng công cụ và phần mềm hỗ trợ

- **Mô tả**: Sử dụng các phần mềm quản lý dự án (MS Project, Jira, Asana) để lập kế hoạch và theo dõi nguồn lực hiệu quả.
- **Áp dụng**: Lên lịch biểu chi tiết, phân công nhiệm vụ và theo dõi hiệu suất làm việc.
- **Lợi ích**: Tăng tính chính xác và giảm sai sót trong quá trình phân bổ nguồn lực.

Ví dụ: Dùng MS Project để theo dõi việc phân bổ nguồn lực và cập nhật tiến độ dự án hàng tuần.

Tóm tắt các nguyên tắc

Nguyên tắc	**Lợi ích**
Xác định rõ mục tiêu và yêu cầu	Tập trung nguồn lực vào các công việc quan trọng.
Đánh giá mức độ quan trọng và tác động	Phân bổ nguồn lực hợp lý theo ưu tiên.

Nguyên tắc	Lợi ích
Xác định ràng buộc về nguồn lực	Sử dụng nguồn lực hiệu quả, tránh quá tải.
Xây dựng tiêu chí ưu tiên	Đưa ra quyết định nhất quán và rõ ràng.
Đảm bảo sự cân bằng	Giảm thiểu kiệt sức và tối ưu hóa năng suất.
Tích hợp các rủi ro	Tăng tính linh hoạt và khả năng ứng phó.
Đánh giá lại và điều chỉnh	Đảm bảo nguồn lực được phân bổ phù hợp theo thời gian.
Sử dụng công cụ hỗ trợ	Tăng hiệu quả và độ chính xác.

Các phương pháp điều chỉnh nguồn lực dự án (chỉnh đều nguồn lực, thời gian dự trữ tối thiểu, hạn chế số lượng nguồn lực)

Điều chỉnh nguồn lực dự án là quá trình phân bổ và tối ưu hóa nguồn lực sao cho phù hợp với tiến độ và yêu cầu công việc, đồng thời hạn chế lãng phí hoặc quá tải. Dưới đây là các phương pháp phổ biến để điều chỉnh nguồn lực trong quản lý dự án:

---

1. Phương pháp chỉnh đều nguồn lực (Resource Leveling)

Mô tả:

- Chỉnh đều nguồn lực là quá trình sắp xếp lại lịch trình dự án sao cho sử dụng nguồn lực đồng đều, tránh tình trạng quá tải hoặc nhàn rỗi.
- Dự án có thể kéo dài thêm thời gian nếu cần đảm bảo nguồn lực không bị vượt quá giới hạn.

Áp dụng:

- Điều chỉnh ngày bắt đầu/kết thúc của các công việc không bắt buộc phải hoàn thành đúng hạn.
- Xem xét các hoạt động có độ linh hoạt cao trong thời gian thực hiện.

Ưu điểm:

- Giảm áp lực cho các nguồn lực như nhân sự, thiết bị, hoặc vật liệu.
- Tránh rủi ro kiệt sức và tăng hiệu suất làm việc.

Hạn chế:

- Có thể làm tăng thời gian hoàn thành dự án.
- Không phù hợp với các dự án có thời hạn nghiêm ngặt.

Ví dụ:

Nếu một nhân viên phải làm hai nhiệm vụ cùng lúc, có thể dời một nhiệm vụ sang thời gian sau để tránh tình trạng làm việc quá sức.

---

2. Phương pháp sử dụng thời gian dự trữ tối thiểu (Critical Chain Project Management - CCPM)

Mô tả:

- Phương pháp này tập trung vào việc tối ưu hóa thời gian dự trữ trong dự án để đảm bảo sử dụng nguồn lực hiệu quả nhất.
- Thời gian dự trữ được phân bổ ở các công việc quan trọng và trong các giai đoạn cần thiết, thay vì trải đều cho mọi nhiệm vụ.

Áp dụng:

- Xác định chuỗi công việc quan trọng nhất (Critical Chain) của dự án.
- Thêm "buffer time" (thời gian dự trữ) vào cuối chuỗi công việc quan trọng hoặc các điểm giao cắt giữa các nhóm nhiệm vụ.

Ưu điểm:

- Tăng tính linh hoạt trong việc sử dụng nguồn lực.
- Hạn chế việc trì hoãn toàn bộ dự án do sự cố trong một công việc.

Hạn chế:

- Đòi hỏi sự phân tích và giám sát kỹ lưỡng.
- Khó thực hiện đối với các dự án lớn hoặc phức tạp.

Ví dụ:

Trong một dự án xây dựng, thêm 10 ngày thời gian dự trữ vào công đoạn cuối cùng để xử lý các vấn đề phát sinh mà không ảnh hưởng đến tiến độ chung.

---

3. Phương pháp hạn chế số lượng nguồn lực (Resource Constrained Scheduling - RCS)

Mô tả:

- Phương pháp này giới hạn số lượng nguồn lực có thể sử dụng tại bất kỳ thời điểm nào, đồng thời tối ưu hóa lịch trình dự án dựa trên các ràng buộc nguồn lực.

- Dự án có thể kéo dài hoặc phải giảm phạm vi nếu nguồn lực bị hạn chế.

Áp dụng:

- Xác định mức độ sẵn có của từng loại nguồn lực (nhân sự, thiết bị, ngân sách).
- Sắp xếp công việc theo mức độ ưu tiên cao nhất, đảm bảo không vượt quá giới hạn nguồn lực.

Ưu điểm:

- Thích hợp cho các dự án có nguồn lực khan hiếm hoặc cố định.
- Duy trì cân đối giữa phạm vi công việc và khả năng thực hiện.

Hạn chế:

- Có thể làm tăng thời gian hoàn thành hoặc giảm phạm vi dự án.
- Yêu cầu phân tích kỹ lưỡng và giám sát liên tục.

Ví dụ:

Một nhóm kỹ sư chỉ có 5 người, nên không thể làm đồng thời 6 nhiệm vụ. Phải sắp xếp nhiệm vụ theo thứ tự ưu tiên, thực hiện lần lượt.

4. Điều chỉnh theo ưu tiên nhiệm vụ (Task Prioritization)

Mô tả:

- Trong phương pháp này, các nhiệm vụ được xếp thứ tự ưu tiên theo mức độ quan trọng và tác động đến tiến độ dự án. Các nguồn lực sẽ được phân bổ trước cho các nhiệm vụ quan trọng và nhiệm vụ ít quan trọng hơn sẽ bị hoãn hoặc cắt giảm nếu cần thiết.

Ưu điểm:

- **Tập trung vào nhiệm vụ cốt lõi:** Đảm bảo các nhiệm vụ quan trọng nhất được hoàn thành đúng hạn.
- **Linh hoạt:** Có thể tái phân bổ nguồn lực nhanh chóng khi ưu tiên thay đổi.

Hạn chế:

- **Bỏ qua nhiệm vụ ít quan trọng:** Có thể gây tác động tiêu cực đến chất lượng hoặc các yêu cầu phụ của dự án.
- **Phụ thuộc vào đánh giá ưu tiên:** Sai lệch trong đánh giá mức độ ưu tiên có thể ảnh hưởng đến toàn bộ dự án.

Ứng dụng:

- Thích hợp cho các dự án có yêu cầu thay đổi thường xuyên hoặc khi gặp phải tình trạng khủng hoảng nguồn lực.

- **Ví dụ:** Trong một dự án phần mềm, nếu nguồn lực hạn chế, đội phát triển có thể tập trung vào xây dựng các tính năng quan trọng trước và hoãn việc triển khai các tính năng phụ.

---

5. Điều chỉnh bằng cách thuê ngoài (Outsourcing or Resource Augmentation)

Mô tả:

- Khi nguồn lực nội bộ không đủ, dự án có thể thuê ngoài hoặc tăng cường nguồn lực bằng cách thuê nhân viên hoặc thiết bị từ các nhà cung cấp bên ngoài.
- Phương pháp này giúp đáp ứng nhu cầu ngắn hạn mà không cần mở rộng cơ sở hạ tầng nội bộ.

Ưu điểm:

- **Giảm áp lực cho nguồn lực nội bộ:** Có thể giải quyết ngay lập tức tình trạng thiếu hụt nguồn lực.
- **Linh hoạt:** Dễ dàng điều chỉnh khi yêu cầu dự án thay đổi.

Hạn chế:

- **Tăng chi phí:** Việc thuê ngoài thường đắt đỏ hơn so với sử dụng nguồn lực nội bộ.
- **Phụ thuộc vào bên thứ ba:** Chất lượng và tiến độ có thể bị ảnh hưởng nếu nhà cung cấp không đáng tin cậy.

Ứng dụng:

- Thích hợp cho các dự án cần hoàn thành nhanh chóng hoặc có yêu cầu đặc biệt không thể đáp ứng bằng nguồn lực nội bộ.
- **Ví dụ:** Thuê nhà thầu ngoài để thực hiện kiểm thử bảo mật trong một dự án phát triển phần mềm.

6. Tăng thời gian làm việc hoặc sử dụng ca kíp (Overtime and Shift Work)

Mô tả:

- Tăng thời gian làm việc (làm thêm giờ) hoặc sử dụng ca kíp để đáp ứng tiến độ dự án khi nguồn lực hạn chế.
- Đây là cách tiếp cận nhanh chóng, nhưng không nên áp dụng lâu dài vì có thể ảnh hưởng đến sức khỏe và hiệu suất của đội ngũ.

Ưu điểm:

- **Đáp ứng nhanh:** Giải quyết kịp thời các vấn đề tiến độ ngắn hạn.
- **Không cần mở rộng nguồn lực:** Tránh việc thuê thêm nhân sự hoặc mua thêm thiết bị.

Hạn chế:

- **Gây mệt mỏi:** Làm thêm giờ kéo dài có thể làm giảm năng suất và tăng khả năng xảy ra sai sót.
- **Chi phí cao:** Chi phí làm thêm giờ thường cao hơn chi phí làm việc thông thường.

Ứng dụng:

- Thích hợp cho các giai đoạn cao điểm hoặc khi cần hoàn thành nhiệm vụ quan trọng trong thời gian ngắn.
- **Ví dụ:** Đội phát triển phần mềm làm thêm giờ để kịp phát hành sản phẩm đúng hạn.

7. Phương pháp sử dụng thời gian dự trữ tối thiểu (Critical Chain Project Management - CCPM)

Mô tả:

- Phương pháp này tập trung vào việc tối ưu hóa thời gian dự trữ trong dự án để đảm bảo sử dụng nguồn lực hiệu quả nhất.
- Thời gian dự trữ được phân bổ ở các công việc quan trọng và trong các giai đoạn cần thiết, thay vì trải đều cho mọi nhiệm vụ.

Áp dụng:

- Xác định chuỗi công việc quan trọng nhất (Critical Chain) của dự án.
- Thêm "buffer time" (thời gian dự trữ) vào cuối chuỗi công việc quan trọng hoặc các điểm giao cắt giữa các nhóm nhiệm vụ.

Ưu điểm:

- Tăng tính linh hoạt trong việc sử dụng nguồn lực.
- Hạn chế việc trì hoãn toàn bộ dự án do sự cố trong một công việc.

Hạn chế:

- Đòi hỏi sự phân tích và giám sát kỹ lưỡng.
- Khó thực hiện đối với các dự án lớn hoặc phức tạp.

Ví dụ:

Trong một dự án xây dựng, thêm 10 ngày thời gian dự trữ vào công đoạn cuối cùng để xử lý các vấn đề phát sinh mà không ảnh hưởng đến tiến độ chung.

So sánh các phương pháp

Phương pháp	Ưu điểm	Hạn chế	Phù hợp với
Chỉnh đều nguồn lực	Giảm áp lực, tăng hiệu suất làm việc	Có thể kéo dài thời gian hoàn thành dự án	Dự án có mức độ linh hoạt về thời gian
Thời gian dự trữ tối thiểu	Giảm thiểu ảnh hưởng từ các vấn đề phát sinh	Đòi hỏi phân tích kỹ lưỡng	Dự án có các chuỗi công việc quan trọng rõ ràng
Hạn chế số lượng nguồn lực	Phù hợp với nguồn lực khan hiếm hoặc cố định	Kéo dài thời gian hoàn thành hoặc giảm phạm vi	Dự án bị giới hạn bởi ngân sách hoặc nhân sự

Các công cụ quản lý chất lượng dự án

Quản lý chất lượng dự án là một phần quan trọng trong quản lý dự án nhằm đảm bảo sản phẩm hoặc dịch vụ đáp ứng các tiêu chuẩn, yêu cầu của khách hàng và các bên liên quan. Dưới đây là các công cụ quản lý chất lượng phổ biến:

---

1. Biểu đồ Pareto (Pareto Chart)

- **Mô tả**: Một loại biểu đồ thanh dùng để xác định các nguyên nhân chính gây ra vấn đề (theo nguyên tắc 80/20: 80% vấn đề xuất phát từ 20% nguyên nhân chính).
- **Ứng dụng**: Xác định các vấn đề quan trọng nhất cần ưu tiên giải quyết trong dự án.
- **Lợi ích**:
    - Giúp tập trung nguồn lực vào các vấn đề quan trọng nhất.
    - Dễ hiểu, trực quan.

Ví dụ: Trong dự án phần mềm, biểu đồ Pareto có thể cho thấy 80% lỗi đến từ 20% các module.

---

2. Biểu đồ kiểm soát (Control Chart)

- **Mô tả**: Một công cụ đồ họa để theo dõi sự thay đổi trong quá trình dựa trên các giới hạn kiểm soát đã xác định.
- **Ứng dụng**: Giám sát tính ổn định và dự đoán khả năng của quy trình trong sản xuất hoặc triển khai dự án.
- **Lợi ích**:
    - Phát hiện sớm các bất thường trong quá trình.
    - Đảm bảo quy trình luôn trong phạm vi kiểm soát.

Ví dụ: Dùng để theo dõi thời gian xử lý lỗi phần mềm qua các chu kỳ phát triển.

3. Biểu đồ nhân quả (Cause-and-Effect Diagram - Fishbone Diagram)

- **Mô tả**: Một biểu đồ hình xương cá giúp xác định và phân tích nguyên nhân gốc rễ của vấn đề.
- **Ứng dụng**: Phân tích sâu các nguyên nhân tiềm ẩn của lỗi hoặc thất bại trong dự án.
- **Lợi ích**:
    - Cung cấp cái nhìn toàn diện về các yếu tố ảnh hưởng.
    - Hỗ trợ việc tìm ra giải pháp cải thiện hiệu quả.

Ví dụ: Phân tích nguyên nhân gây ra chậm trễ trong tiến độ dự án xây dựng.

---

4. Bảng kiểm tra (Checklist)

- **Mô tả**: Một danh sách các mục tiêu hoặc tiêu chí để đảm bảo mọi công việc đều được thực hiện theo kế hoạch.
- **Ứng dụng**: Kiểm tra các yêu cầu kỹ thuật, tiêu chuẩn chất lượng hoặc hoàn thành công việc cụ thể.
- **Lợi ích**:
    - Đơn giản, dễ thực hiện.
    - Giảm thiểu sai sót và bỏ sót công việc.

Ví dụ: Lập bảng kiểm tra các bước cần thực hiện trước khi triển khai hệ thống phần mềm.

---

5. Biểu đồ phân tán (Scatter Diagram)

- **Mô tả**: Một biểu đồ để phân tích mối quan hệ giữa hai biến số (ví dụ: hiệu suất và thời gian).
- **Ứng dụng**: Phát hiện các mối quan hệ tiềm năng giữa các yếu tố trong quy trình dự án.
- **Lợi ích**:
    - Hỗ trợ phân tích dữ liệu và xác định xu hướng.
    - Dễ dàng áp dụng trong phân tích chất lượng.

Ví dụ: Phân tích mối quan hệ giữa số giờ làm thêm và tỷ lệ lỗi phát sinh.

---

6. Lưu đồ (Flowchart)

- **Mô tả**: Một biểu đồ hình ảnh hóa các bước trong một quy trình hoặc công việc cụ thể.

- **Ứng dụng**: Hiểu và tối ưu hóa quy trình công việc, phát hiện các bước không cần thiết hoặc chồng chéo.
- **Lợi ích**:
    - Tăng tính minh bạch và rõ ràng cho quy trình.
    - Giảm thiểu sai sót trong thực hiện.

Ví dụ: Lưu đồ mô tả quy trình kiểm tra chất lượng sản phẩm trước khi giao hàng.

---

7. Six Sigma Tools (DMAIC và DMADV)

- **Mô tả**:
    - DMAIC: Define, Measure, Analyze, Improve, Control (xác định, đo lường, phân tích, cải tiến, kiểm soát).
    - DMADV: Define, Measure, Analyze, Design, Verify (xác định, đo lường, phân tích, thiết kế, xác minh).
- **Ứng dụng**: Cải thiện và thiết kế quy trình để đạt mức chất lượng cao nhất.
- **Lợi ích**:
    - Tăng hiệu quả và độ chính xác của quy trình.
    - Giảm thiểu lỗi và sai sót.

Ví dụ: Áp dụng DMAIC để cải thiện tốc độ sản xuất trong nhà máy.

---

8. Phân tích FMEA (Failure Mode and Effects Analysis)

- **Mô tả**: Công cụ phân tích rủi ro để xác định các chế độ hỏng hóc tiềm năng và tác động của chúng.
- **Ứng dụng**: Đánh giá mức độ nghiêm trọng, tần suất xảy ra và khả năng phát hiện lỗi trong dự án.
- **Lợi ích**:
    - Ngăn ngừa lỗi trước khi chúng xảy ra.
    - Tăng độ tin cậy của quy trình và sản phẩm.

Ví dụ: Phân tích FMEA trong thiết kế ô tô để ngăn ngừa lỗi động cơ.

---

9. Biểu đồ cột (Histogram)

- **Mô tả**: Một biểu đồ thể hiện phân bố dữ liệu, giúp phân tích tần suất của các giá trị hoặc lỗi.

- **Ứng dụng**: Xác định các xu hướng hoặc kiểu mẫu trong dữ liệu chất lượng.
- **Lợi ích**:
    - Phân tích dễ dàng và trực quan.
    - Hỗ trợ ra quyết định dựa trên dữ liệu.

Ví dụ: Phân tích số lượng lỗi xảy ra ở từng giai đoạn của dự án.

---

10. Kiểm toán chất lượng (Quality Audit)

- **Mô tả**: Quy trình đánh giá độc lập để đảm bảo dự án tuân thủ các tiêu chuẩn và quy trình chất lượng đã đặt ra.
- **Ứng dụng**: Kiểm tra định kỳ để phát hiện và khắc phục các vấn đề về chất lượng.
- **Lợi ích**:
    - Đảm bảo tuân thủ quy định và tiêu chuẩn.
    - Cải thiện liên tục các quy trình.

Ví dụ: Kiểm toán chất lượng trong dự án xây dựng để đảm bảo vật liệu đạt tiêu chuẩn.

11. Sơ đồ xương cá (Cause-and-Effect Diagram hoặc Ishikawa Diagram)

Mô tả:

- Sơ đồ xương cá là một công cụ xác định nguyên nhân gốc rễ của một vấn đề cụ thể. Các nguyên nhân được nhóm theo các yếu tố như con người, quy trình, máy móc, vật liệu, môi trường.

Ưu điểm:

- **Phân tích nguyên nhân sâu:** Giúp nhóm dự án hiểu rõ các yếu tố dẫn đến vấn đề.
- **Trực quan:** Dễ dàng trình bày và sử dụng trong các cuộc họp.

Hạn chế:

- **Không đưa ra giải pháp:** Chỉ xác định nguyên nhân mà không đề xuất cách khắc phục.
- **Đòi hỏi sự tham gia:** Cần nhiều thông tin và ý kiến từ các bên liên quan.

Ứng dụng:

- Xác định nguyên nhân gốc rễ của các lỗi phần mềm hoặc sự cố trong sản xuất.

12. Lưu đồ quy trình (Flowcharts)

Mô tả:

- Lưu đồ quy trình là một sơ đồ trực quan hiển thị các bước trong một quy trình, từ đầu vào đến đầu ra, với các mối quan hệ giữa các bước.

Ưu điểm:

- **Dễ hiểu:** Giúp nhóm dự án nắm bắt rõ quy trình làm việc.
- **Phát hiện vấn đề:** Dễ dàng phát hiện các điểm nghẽn hoặc bước không cần thiết.

Hạn chế:

- **Phụ thuộc vào người tạo:** Nếu không được thiết kế kỹ lưỡng, lưu đồ có thể không đầy đủ hoặc gây nhầm lẫn.
- **Không phân tích nguyên nhân:** Chỉ hiển thị các bước mà không chỉ rõ nguyên nhân vấn đề.

Ứng dụng:

- Thiết kế và cải tiến quy trình phát triển phần mềm, quy trình kiểm thử.

13. Phân tích SWOT (SWOT Analysis)

Mô tả:

- Công cụ này phân tích các yếu tố bên trong (Điểm mạnh - Strengths, Điểm yếu - Weaknesses) và bên ngoài (Cơ hội - Opportunities, Thách thức - Threats) ảnh hưởng đến chất lượng dự án.

Ưu điểm:

- **Phân tích toàn diện:** Cung cấp cái nhìn tổng quan về các yếu tố ảnh hưởng đến chất lượng.
- **Linh hoạt:** Áp dụng được cho nhiều loại dự án.

Hạn chế:

- **Đòi hỏi thông tin:** Kết quả phụ thuộc vào chất lượng dữ liệu đầu vào.
- **Không cụ thể:** Không đưa ra các giải pháp chi tiết.

Ứng dụng:

- Xác định các yếu tố ảnh hưởng đến sự thành công của dự án phát triển phần mềm hoặc thiết kế sản phẩm.

14. Ma trận kiểm tra (Check Sheets)

Mô tả:

- Ma trận kiểm tra là một công cụ thu thập và tổ chức dữ liệu theo thời gian hoặc theo danh mục, giúp xác định xu hướng, mô hình và các yếu tố lặp lại.

Ưu điểm:

- **Dễ sử dụng:** Công cụ đơn giản để thu thập và phân tích dữ liệu.
- **Theo dõi chi tiết:** Hỗ trợ phát hiện các vấn đề thường xuyên xảy ra.

Hạn chế:

- **Chỉ cung cấp dữ liệu thô:** Cần các công cụ khác để phân tích chi tiết.
- **Không phù hợp cho các vấn đề phức tạp:** Chỉ dùng cho các vấn đề cụ thể, đơn giản.

Ứng dụng:

- Thu thập dữ liệu lỗi phần mềm, thống kê số lần trễ thời hạn trong dự án.

15. Phân tích 5 Why (5 Whys Analysis)

Mô tả:

- Phương pháp 5 Why tập trung vào việc hỏi "Tại sao?" liên tiếp năm lần (hoặc nhiều hơn) để tìm ra nguyên nhân gốc rễ của một vấn đề.

Ưu điểm:

- **Đơn giản:** Không cần công cụ phức tạp.
- **Tìm nguyên nhân gốc rễ:** Tập trung giải quyết vấn đề từ căn bản.

Hạn chế:

- **Phụ thuộc vào chất lượng câu hỏi:** Nếu không hỏi đúng câu hỏi, nguyên nhân gốc có thể không được xác định.
- **Chủ quan:** Kết quả có thể phụ thuộc vào người thực hiện.

Ứng dụng:

- Xác định lý do chậm tiến độ trong một dự án hoặc nguyên nhân lỗi sản phẩm.

16. Biểu đồ Histogram

Mô tả:

- Biểu đồ Histogram thể hiện dữ liệu tần suất (frequency data) của một tập hợp, giúp xác định xu hướng hoặc phân bố.

Ưu điểm:

- **Phân tích dữ liệu dễ dàng:** Dễ dàng nhận biết các xu hướng hoặc bất thường.
- **Trực quan:** Dễ hiểu ngay cả với người không chuyên.

Hạn chế:

- **Không xác định nguyên nhân:** Chỉ trình bày dữ liệu mà không chỉ ra nguyên nhân.
- **Không phù hợp cho dữ liệu nhỏ:** Cần tập hợp dữ liệu lớn để tạo ra biểu đồ đáng tin cậy.

Ứng dụng:

- Phân tích các lỗi thường gặp trong kiểm thử phần mềm hoặc đánh giá độ ổn định của quy trình sản xuất.

17. Kiểm soát danh sách kiểm tra (Quality Checklists)

Mô tả:

- Danh sách kiểm tra là một công cụ đơn giản để đảm bảo rằng tất cả các bước, tiêu chuẩn, hoặc yêu cầu chất lượng được hoàn thành.

Ưu điểm:

- **Dễ sử dụng:** Phù hợp với các nhóm nhỏ hoặc cá nhân.
- **Giảm sai sót:** Đảm bảo không bỏ sót các bước quan trọng.

Hạn chế:

- **Không linh hoạt:** Nếu không được cập nhật thường xuyên, danh sách kiểm tra có thể trở nên lỗi thời.
- **Không phù hợp với các vấn đề phức tạp:** Chỉ phù hợp cho các tác vụ đơn giản.

Ứng dụng:

- Sử dụng danh sách kiểm tra trong kiểm thử phần mềm để đảm bảo tất cả các tính năng đã được kiểm tra.

18. Biểu đồ phân tán (Scatter Diagram)

- **Mô tả**: Một biểu đồ để phân tích mối quan hệ giữa hai biến số (ví dụ: hiệu suất và thời gian).
- **Ứng dụng**: Phát hiện các mối quan hệ tiềm năng giữa các yếu tố trong quy trình dự án.
- **Lợi ích**:
    - Hỗ trợ phân tích dữ liệu và xác định xu hướng.
    - Dễ dàng áp dụng trong phân tích chất lượng.

Ví dụ: Phân tích mối quan hệ giữa số giờ làm thêm và tỷ lệ lỗi phát sinh.

Tóm tắt các công cụ

Công cụ	**Ứng dụng chính**	**Lợi ích chính**
Biểu đồ Pareto	Xác định nguyên nhân chính gây vấn đề	Ưu tiên giải quyết các vấn đề quan trọng
Biểu đồ kiểm soát	Theo dõi tính ổn định của quy trình	Phát hiện sớm bất thường
Biểu đồ nhân quả	Phân tích nguyên nhân gốc rễ	Tìm giải pháp cải thiện hiệu quả
Bảng kiểm tra	Kiểm tra công việc, yêu cầu	Đơn giản, dễ thực hiện

Công cụ	Ứng dụng chính	Lợi ích chính
Biểu đồ phân tán	Phân tích mối quan hệ giữa các yếu tố	Xác định xu hướng
Lưu đồ	Hình ảnh hóa quy trình	Tăng minh bạch và giảm sai sót
Six Sigma	Cải thiện và thiết kế quy trình	Đạt chất lượng cao nhất
FMEA	Phân tích và ngăn ngừa lỗi	Tăng độ tin cậy
Biểu đồ cột (Histogram)	Phân tích tần suất lỗi	Hiểu rõ phân bố dữ liệu
Kiểm toán chất lượng	Đánh giá sự tuân thủ tiêu chuẩn	Đảm bảo chất lượng

Thực hành (4.0 điểm), 02 CÂU

1. Tính độ đo theo chức năng (Function Point): Sinh viên được cung cấp bảng dữ liệu các về các tham biến độ đo (Measurement Parameters) và giá trị nhân tố trọng số (Weighing Factor), các giá trị của 14 questions để tính toán FP.

CAF = [0.65 + 0.01 * ∑(fi)]

2. Tìm đường găng, tìm ngày bắt đầu sớm (ES), kết thúc sớm (EF), bắt đầu trễ (LS), kết thúc trễ (LF), thời gian dự trữ theo sơ đồ Pert cho sẵn.

3. Cho bảng danh sách gồm tên công việc, thời gian thực hiện, số lượng nhân sự thực hiện. SV xây dựng sơ đồ PERT, tìm đường găng và vẽ sơ đồ phụ tải nguồn lực, sơ đồ điều chỉnh nguồn lực theo hướng hạn chế nhân lực nhất.

Một dự án phát triển phần mềm được giao với các yêu cầu như sau:

1. **External Inputs (EI)**: 10
2. **External Outputs (EO)**: 5
3. **External Inquiries (EQ)**: 8
4. **Internal Logical Files (ILF)**: 4
5. **External Interface Files (EIF)**: 3

Bảng giá trị **Weighing Factor** cho từng tham biến:

Tham biến	Đơn giản (Simple)	Trung bình (Average)	Phức tạp (Complex)

EI	3	4	6
EO	4	5	7
EQ	3	4	6
ILF	7	10	15
EIF	5	7	10

Dự án có các tham biến được đánh giá như sau:

- **EI**: 6 đơn giản, 4 trung bình.

  (**6 đơn giản (Simple)**: ví dụ: nhận đầu vào từ người dùng mà không yêu cầu tính toán phức tạp hoặc xác thực).

  4 trung bình (Average): ví dụ: cần xác thực hoặc liên kết dữ liệu từ cơ sở dữ liệu.)

- **EO**: 2 trung bình, 3 phức tạp.
- **EQ**: 5 đơn giản, 3 trung bình.
- **ILF**: 3 trung bình, 1 phức tạp.
- **EIF**: 2 đơn giản, 1 phức tạp.

CAF (Complexity Adjustment Factor) được tính theo công thức:

$$CAF = 0.65 + 0.01 \times \sum_{i=1}^{14} f_i$$

Với f_i là giá trị từ 14 câu hỏi về các yếu tố phức tạp của hệ thống. Giá trị f_i cho từng câu hỏi được cung cấp như sau:

{4, 3, 4, 5, 3, 4, 4, 2, 5, 3, 4, 3, 4, 3}.

- $f_i = 0$: Không có yếu tố phức tạp.
- $f_i = 5$: Hệ thống có mức độ phức tạp rất cao ở khía cạnh này.

Trong dữ liệu được cung cấp:

- Bộ giá trị $f_i = \{4,3,4,5,3,4,4,2,5,3,4,3,4,3\}$ là **đánh giá cho từng câu hỏi**:

1. Câu hỏi 1: Được đánh giá là **4** → Mức độ phức tạp cao.
2. Câu hỏi 2: Được đánh giá là **3** → Mức độ phức tạp trung bình.
3. …

4. Câu hỏi 14: Được đánh giá là **3** → Mức độ phức tạp trung bình.

Các giá trị này sẽ được **cộng lại** để tính tổng $\sum f_i$, được sử dụng để xác định **Complexity Adjustment Factor (CAF)**.

Yêu cầu:

1. Tính tổng điểm không điều chỉnh (Unadjusted Function Point - UFP).
2. Tính giá trị CAF.
3. Tính tổng Function Point (FP) của hệ thống.

Bước 1: Tính tổng điểm không điều chỉnh (UFP)

UFP được tính theo công thức:

$$UFP = \sum (Số lượng \times Trọng số)$$

- $\boldsymbol{EI}$: $6 \times 3 + 4 \times 4 = 18 + 16 = 34$
- $\boldsymbol{EO}$: $2 \times 5 + 3 \times 7 = 10 + 21 = 31$
- $\boldsymbol{EQ}$: $5 \times 3 + 3 \times 4 = 15 + 12 = 27$
- $\boldsymbol{ILF}$: $3 \times 10 + 1 \times 15 = 30 + 15 = 45$
- $\boldsymbol{EIF}$: $2 \times 5 + 1 \times 10 = 10 + 10 = 20$

Tính toán trọng số:

- Dựa trên số lượng EI được đánh giá trong dự án:
- **Tổng trọng số cho EI đơn giản**: $6 \times 3 = 18$.
- **Tổng trọng số cho EI trung bình**: $4 \times 4 = 16$.
- Tổng trọng số của tất cả các $EI = 18 + 16 = 34$.

Tổng UFP:

$$\text{UFP} = 34 + 31 + 27 + 45 + 20 = 157$$

Bước 2: Tính giá trị CAF

$$CAF = 0.65 + 0.01 \times \sum_{i=1}^{14} f_i$$

$$\text{UFP} = 34 + 31 + 27 + 45 + 20 = 157$$

Tính tổng các giá trị f_i :

$$\sum_{i=1}^{14} f_i = 4 + 3 + 4 + 5 + 3 + 4 + 4 + 2 + 5 + 3 + 4 + 3 + 4 + 3 = 51$$

Thay vào công thức CAF:

$$CAF = 0.65 + 0.01 \times 51 = 0.65 + 0.51 = 1.16$$

Bước 3: Tính tổng Function Point (FP)

$$FP = UFP \times CAF$$

Thay giá trị:

$$FP = 157 \times 1.16 = 182.12$$

Bài tiếp theo: Một công ty cần thực hiện dự án phát triển phần mềm trong vòng 6 tháng. Dự án bao gồm các công việc như sau:

Tên công việc	Thời gian (ngày)	Số nhân sự thực hiện	Công việc phụ thuộc
A	5	2	Không
B	7	3	A
C	10	2	B
D	12	4	C
E	8	3	D
F	6	2	E

Yêu cầu:

1. **Vẽ sơ đồ PERT**: Biểu diễn các công việc, thời gian và mối liên kết phụ thuộc giữa các công việc.
2. **Tìm đường găng (Critical Path)**: Xác định đường găng của dự án và thời gian hoàn thành.
3. **Sơ đồ phụ tải nguồn lực**: Vẽ biểu đồ thể hiện việc phân bổ nguồn lực theo thời gian.
4. **Sơ đồ điều chỉnh nguồn lực**: Điều chỉnh sơ đồ phân bổ nguồn lực sao cho hạn chế số lượng nhân sự tối đa cần thiết tại một thời điểm.

Xác định công việc phụ thuộc:

Công việc A (Thu thập yêu cầu) bắt đầu đầu tiên.

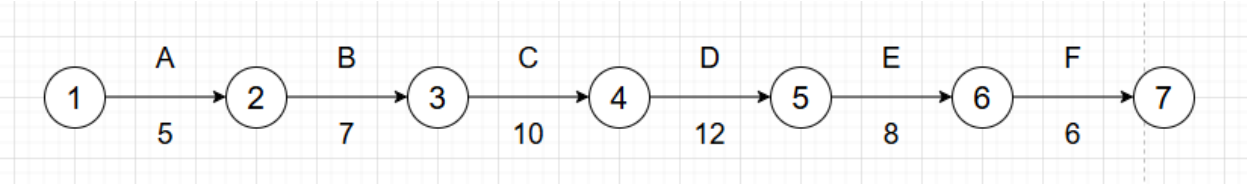
Công việc B (Phân tích yêu cầu) phụ thuộc vào **A**.

Công việc C (Thiết kế hệ thống) phụ thuộc vào **B**.

Công việc D (Lập trình module) phụ thuộc vào **C**.

Công việc E (Kiểm thử và sửa lỗi) phụ thuộc vào **D**.

Công việc F (Triển khai hệ thống) phụ thuộc vào **E**.



Thời gian sớm nhất (Early Start - ES)

và hoàn thành sớm nhất (Early Finish - EF):

A: **ES = 0**, **EF = 5**.

B: **ES = 5**, **EF = 12**.

C: **ES = 12**, **EF = 22**.

D: **ES = 22**, **EF = 34**.

E: **ES = 34**, **EF = 42**.

F: **ES = 42**, **EF = 48**.

Thời gian muộn nhất (Late Start - LS) và hoàn thành muộn nhất (Late Finish - LF):

F: **LF = 48**, **LS = 42**.

E: **LF = 42**, **LS = 34**.

D: **LF = 34**, **LS = 22**.

C: **LF = 22**, **LS = 12**.

B: **LF = 12**, **LS = 5**.

A: **LF = 5**, **LS = 0**.

Slack (Độ chậm trễ):

A: **Slack = 0**.

B: **Slack = 0**.

C: **Slack = 0**.

D: **Slack = 0**.

E: **Slack = 0**.

F: **Slack = 0**.

Thời gian hoàn thành dự án: **48 ngày**(A --> B --> C --> D --> E --> F)

3. Sơ đồ phụ tải nguồn lực

Phân bổ nguồn lực theo từng công việc:

Ngày 1-5: 2 nhân sự cho công việc A.

Ngày 6-12: 3 nhân sự cho công việc B.

Ngày 13-22: 2 nhân sự cho công việc C.

Ngày 23-34: 4 nhân sự cho công việc D.

Ngày 35-42: 3 nhân sự cho công việc E.

Ngày 43-48: 2 nhân sự cho công việc F.

Ngày	**Nhân sự**
1-5	2
6-12	3
13-22	2
23-34	4
35-42	3
43-48	2

4. Sơ đồ điều chỉnh nguồn lực

Để giảm số lượng nhân sự tối đa trong cùng thời điểm:

Chia công việc **D** (Lập trình module) thành các giai đoạn nhỏ hơn hoặc chuyển một phần nhân sự từ công việc khác sang khi có thể.

Điều chỉnh:

Kéo dài công việc **D** từ ngày **23-34** thành ngày **23-38**, sử dụng 3 nhân sự thay vì 4.

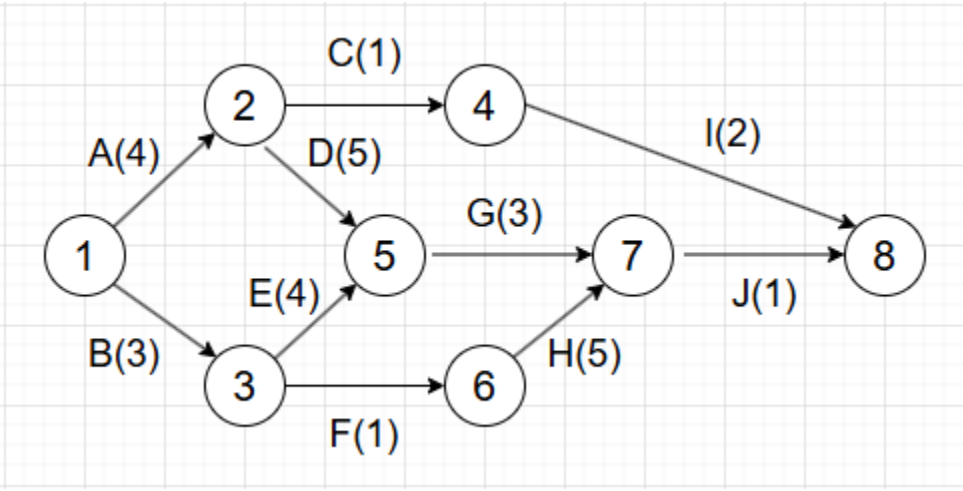
Ngày	Nhân sự
1-5	2
6-12	3
13-22	2
23-38	3
39-42	3
43-48	2

Bài 2:

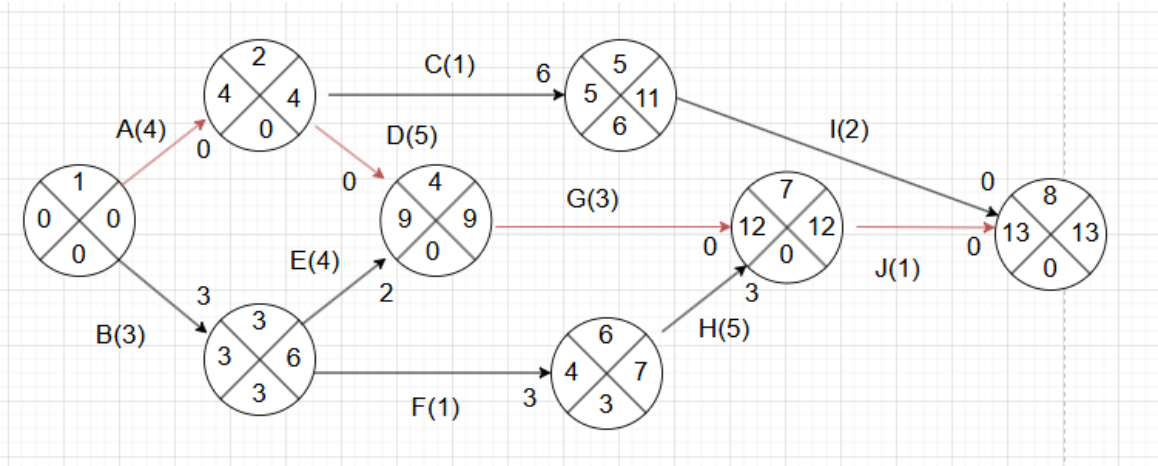
CV	CV trước	Thời gian (ngày)
A		4
B		3
C	A	1
D	A	5
E	B	4
F	B	1
G	D, E	3
H	F	5
I	C	2
J	G, H	1

a) Vẽ sơ đồ Pert sau, SV xác định đường găng.
b) Vẽ sơ đồ phụ tải nguồn lực
c) Điều chỉnh sơ đồ phụ tải nguồn lực theo hướng thời gian dự trữ tối thiểu.

a) Pert



AOA:



Thời gian sớm nhất (Early Start - ES)

và hoàn thành sớm nhất (Early Finish - EF):

A: **ES = 0**, **EF = 4**.

B: **ES = 0**, **EF = 3**.

C: **ES = 4**, **EF = 5**.

D: **ES = 4**, **EF = 9**.

E: **ES = 3**, **EF = 7**.

F: **ES = 3**, **EF = 4**.

G: **ES = 9**, **EF = 12**.

H: **ES = 4**, **EF = 9**.

I: **ES = 5**, **EF = 7**.

J: **ES = 12**, **EF = 13**.

Thời gian muộn nhất (Late Start - LS) và hoàn thành muộn nhất (Late Finish - LF):

J: **LF = 13**, **LS = 12**.

I: **LF = 13**, **LS = 11**.

H: **LF = 12**, **LS = 7**.

G: **LF = 12**, **LS = 9**.

F: **LF = 7**, **LS = 6**.

E: **LF =9** , **LS = 5** .

D: **LF = 9**, **LS = 4**.

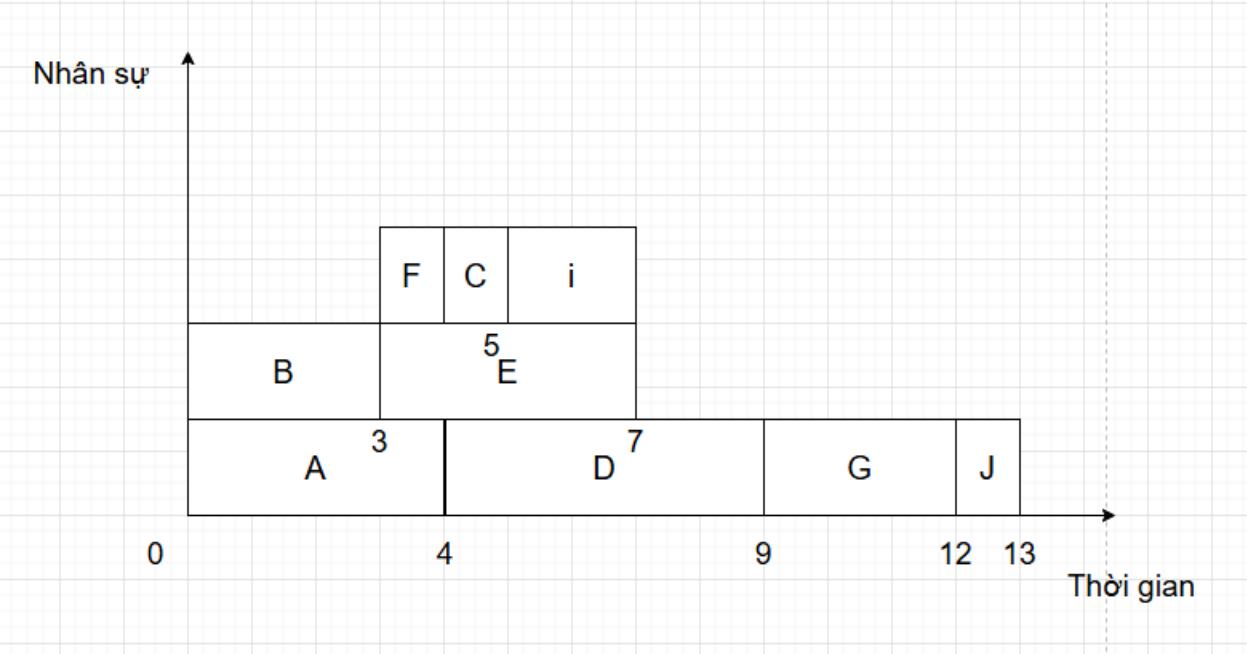
C: **LF = 11**, **LS = 10** .

B: **LF = 5** , **LS = 2** .

A: **LF = 4** , **LS = 0**.

Gang => A-D-G-J tổng là 13 ngày

b): sơ đồ phụ tải nguồn lực



c): điều chỉnh phụ tải nguồn lực

Để giảm số lượng nhân sự tối đa trong cùng thời điểm:

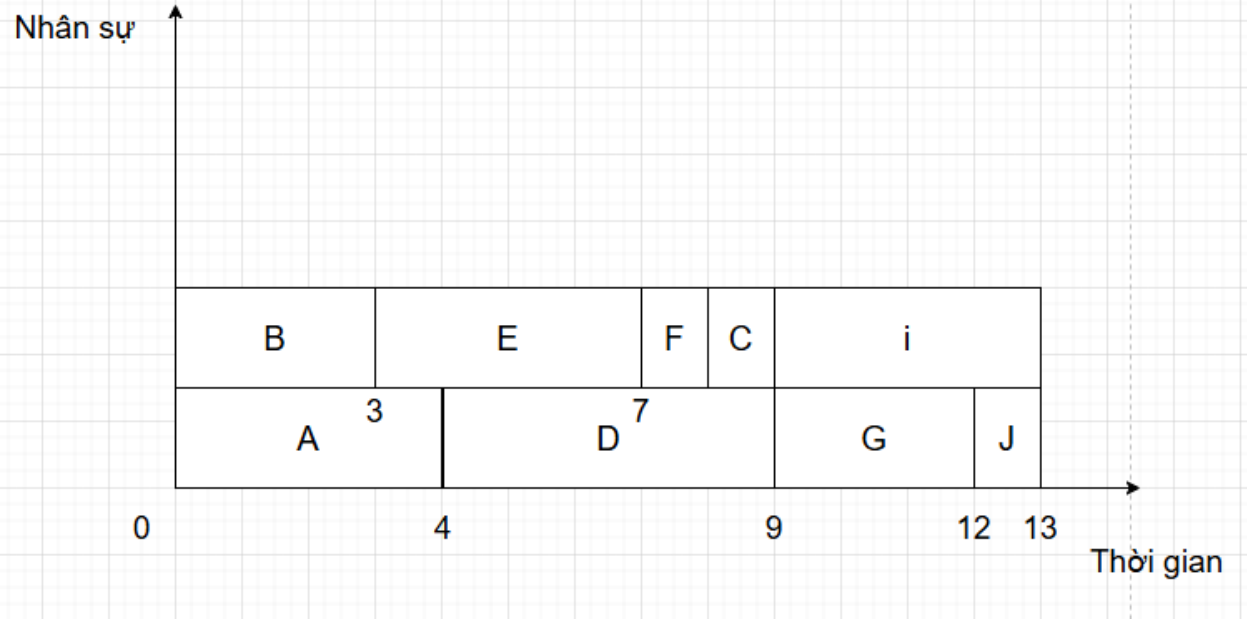
Chia công việc **I**(Lập trình module) thành các giai đoạn nhỏ hơn hoặc chuyển một phần nhân sự từ công việc khác sang khi có thể.

Điều chỉnh:

Giảm công việc **I** từ ngày **5-7** thành ngày **9-13**, sử dụng 3 nhân sự thay vì 4.

Công việc **I** từ ngày **3-4** thành ngày **7-8**, sử dụng 3 nhân sự thay vì 4.

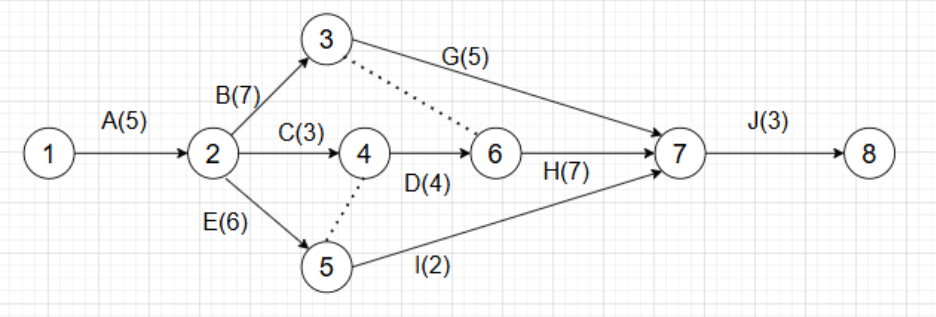
Công việc **C** từ ngày **4-5** thành ngày **8-9**, sử dụng 3 nhân sự thay vì 4.



Bài 3:

A	-	5
B	A	7
C	A	3
D	C	4
E	A	6
G	B	5
H	D,B	7
I	E,C	2
J	G,H,I	3

a): sơ đồ Pert



Thời gian sớm nhất (Early Start - ES)

và hoàn thành sớm nhất (Early Finish - EF):

A: **ES = 0**, **EF = 5**.

B: **ES = 5**, **EF = 12**.

C: **ES = 5**, **EF = 8**.

D: **ES = 8**, **EF = 12**.

E: **ES = 5**, **EF = 11**.

G: **ES = 12**, **EF = 17**.

H: **ES = 12**, **EF = 19**.

I: **ES = 11**, **EF = 13**.

J: **ES = 19**, **EF = 22**.

Thời gian muộn nhất (Late Start - LS) và hoàn thành muộn nhất (Late Finish - LF):

J: **LF = 22**, **LS = 19**.

I: **LF = 19**, **LS = 17**.

H: **LF = 19**, **LS = 12**.

G: **LF = 19**, **LS = 14**.

E: **LF =17**, **LS =11** .

D: **LF = 12**, **LS = 8**.

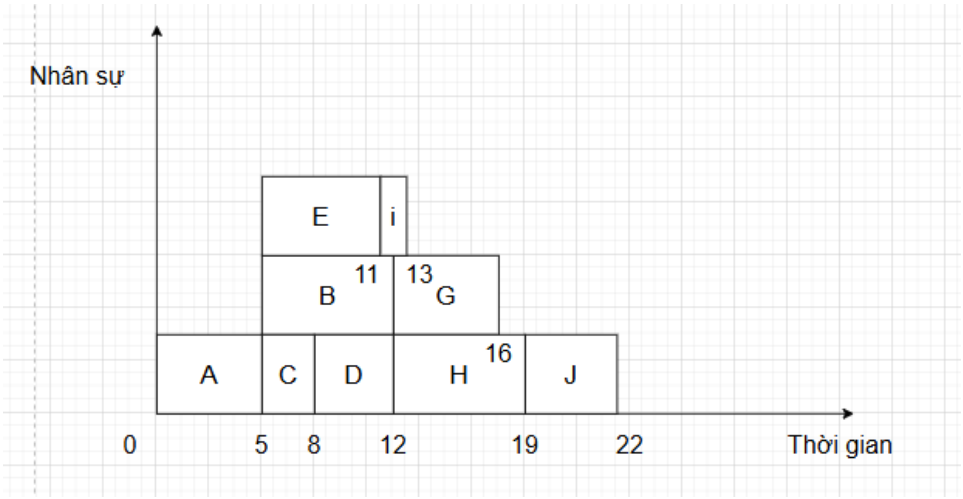
C: **LF = 8**, **LS = 5** .

B: **LF = 14** , **LS = 7** .

A: **LF = 5** , **LS = 0**.

Gang => A-D-C-H-J tổng là 19 ngày

b): biểu đồ phụ tải nguồn lực



c): điều chỉnh phụ tải nguồn lực

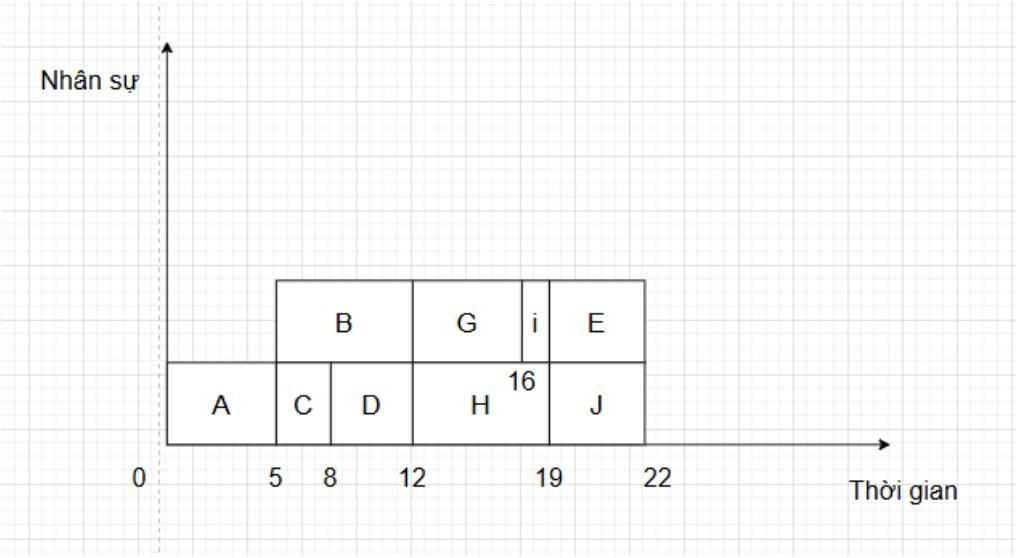
Để giảm số lượng nhân sự tối đa trong cùng thời điểm:

Chia công việc **E** (Lập trình module) thành các giai đoạn nhỏ hơn hoặc chuyển một phần nhân sự từ công việc khác sang khi có thể.

Điều chỉnh:

Công việc **I** từ ngày **11-13** thành ngày **16-19**, sử dụng 3 nhân sự thay vì 4.

Giảm công việc **E** từ ngày **5-11** thành ngày **19-22**, sử dụng 3 nhân sự thay vì 4.



Nếu hoàn thành trong 25 ngày thì khả năng là bao nhiêu phần trăm?

ct phương sai:

$$\sigma^2 = \left[\frac{b-a}{6}\right]^2$$

A-D-C-H-J tổng là 19 ngày

Lưu ý : nếu có 2 đường găng tính tổng phương sai nào lớn thì chọn.

A = 5/6 bình phương = 0,69

D= 0,44

C=0,25

H=1,36

J=0,25

Tổng = 2,99 => độ lệnh chuổn bằng căn bậc 2 =1,72

Vậy % hoàn thành dự án trong 25 ngày là (25-22)/1,72=1,74 là 1,7 và 0,04