## KIỂM THỬ PHẦN MỀM

### Hệ thống nhúng là gì?

Phần mềm nhúng là một chương trình được viết, biên dịch trên máy tính và nạp vào một hệ thống khác bao gồm một hoặc nhiều bộ vi xử lý đã được cài sẵn một hệ điều hành, bộ nhớ ghi chép được, các cổng giao tiếp với các phần cứng khác… Hệ thống nhúng là các thiết bị được điều khiển điện tử trong đó phần mềm và phần cứng được kết hợp chặt chẽ.

### Kiểm thử phần mềm nhúng là gì?

Kiểm thử phần mềm nhúng là kiểm tra các hệ thống nhúng về hiệu suất, tính nhất quán và được xác thực theo yêu cầu của khách hàng và được thực hiện bởi nhóm phát triển phần mềm.

Kiểm tra phần mềm nhúng kiểm tra và đảm bảo phần mềm liên quan có chất lượng tốt và tuân thủ tất cả các yêu cầu cần đáp ứng.

### Mục đích Kiểm thử phần mềm nhúng

Kiểm thử phần mềm nhúng vì bốn lý do:

* Để tìm lỗi trong phần mềm.
* Giúp giảm rủi ro cho cả người dùng và công ty phát triển.
* Cắt giảm chi phí phát triển và bảo trì.
* Để cải thiện hiệu suất.

Để kiểm thử hệ thống nhúng, các hoạt động sau được thực hiện:

1. Cung cấp đầu vào cho phần mềm.
2. Một phần của phần mềm được thực thi.
3. Đánh giá ngõ ra (kiểm tra dữ liệu đầu ra có khớp với kết quả mong đợi hay không, phù hợp với các yêu cầu và không có sự cố hệ thống).

### Các loại kiểm thử phần mềm nhúng

Về cơ bản, có năm cấp độ thử nghiệm có thể được áp dụng cho phần mềm nhúng.

#### Kiểm thử đơn vị phần mềm

Các mô-đun đơn vị là một chức năng hoặc lớp. Kiểm thử đơn vị được thực hiện bởi nhóm phát triển, chủ yếu là nhà phát triển và thường được thực hiện theo mô hình đánh giá ngang hàng. Dựa trên đặc điểm kỹ thuật của các trường hợp thử nghiệm mô-đun được phát triển.

#### Kiểm thử tích hợp

Kiểm thử tích hợp có thể được phân thành hai phân khúc:

Kiểm thử tích hợp phần mềm / phần cứng.

Phát triển phần mềm nhúng có một đặc điểm duy nhất tập trung vào môi trường thực tế. Điều này gây ra sự bất tiện cho thử nghiệm vì thử nghiệm toàn diện không thể được thực hiện trong điều kiện mô phỏng.

#### Kiểm thử đơn vị hệ thống

Mô-đun được kiểm tra là một khung đầy đủ bao gồm mã phần mềm hoàn chỉnh bổ sung tất cả hệ điều hành thời gian thực (RTOS- real- time operating system) và các phần liên quan đến nền tảng như ngắt, cơ chế tác vụ, truyền thông, v.v. Giao thức Point of Control không còn là một cuộc gọi đến một chức năng hoặc một lời gọi phương thức, mà là một tin nhắn được gửi / nhận sử dụng các hàng đợi tin nhắn RTOS.

Tài nguyên hệ thống được quan sát để đánh giá khả năng của hệ thống để hỗ trợ thực thi hệ thống nhúng. Đối với khía cạnh này, thử nghiệm hộp xám là phương pháp thử nghiệm được ưa chuộng. Tùy thuộc vào tổ chức, kiểm tra đơn vị hệ thống là nhiệm vụ của nhà phát triển hoặc nhóm tích hợp hệ thống chuyên dụng.

#### Kiểm thử tích hợp hệ thống

Các điểm kiểm soát và quan sát (PCO- Points of Control and Observations) là sự pha trộn của các giao thức truyền thông và RTOS liên quan đến mạng, như tin nhắn mạng và các sự kiện RTOS.

#### Kiểm tra xác nhận hệ thống

Mô-đun được kiểm tra là một hệ thống con hoặc hệ thống nhúng hoàn chỉnh. Mục tiêu của thử nghiệm cuối cùng này là để đáp ứng các yêu cầu chức năng thực thể bên ngoài. Việc kiểm tra liên quan đến thực thể bên ngoài hoặc một thiết bị trong mạng viễn thông hoặc cả hai.

### Sự khác biệt: Kiểm thử nhúng và Kiểm thử phần mềm

|  |  |
| --- | --- |
| **Kiểm thử phần mềm** | **Kiểm tra nhúng** |
| Kiểm thử phần mềm chỉ liên quan đến phần mềm. | Kiểm tra nhúng có liên quan đến cả phần mềm cũng như phần cứng. |
| Trung bình 90% thử nghiệm được thực hiện trên thế giới hoàn toàn là thử nghiệm hộp đen thủ công. | Thử nghiệm nhúng được thực hiện trên các hệ thống nhúng hoặc chip, nó có thể là thử nghiệm hộp đen hoặc hộp trắng. |
| Các lĩnh vực kiểm tra chính là kiểm tra GUI, chức năng, xác nhận và một số cấp độ kiểm tra cơ sở dữ liệu. | Các lĩnh vực chính của thử nghiệm là hoạt động của phần cứng dựa trên đầu vào |
| Kiểm thử phần mềm được thực hiện chủ yếu trên các ứng dụng máy khách, máy chủ và web. | Kiểm tra nhúng thường được thực hiện trên Phần cứng. |

### Thách thức: Kiểm thử phần mềm nhúng

Một số thách thức có thể gặp phải trong quá trình kiểm thử phần mềm nhúng:

#### Phụ thuộc phần cứng

Sự phụ thuộc phần cứng là một trong những khó khăn chính gặp phải trong quá trình kiểm thử phần mềm nhúng vì quyền truy cập hạn chế vào phần cứng. Tuy nhiên, Trình mô phỏng có thể không thể hiện chính xác hoạt động của thiết bị thực tế và có thể hiểu sai về hiệu suất hệ thống và khả năng sử dụng của ứng dụng.

#### Phần mềm mã nguồn mở

Phần lớn các thành phần phần mềm nhúng là nguồn mở, không có kiểm tra hoàn chỉnh có sẵn cho nó.

#### Lỗi phần mềm so với phần cứng

Một khía cạnh khác là khi phần mềm đang được phát triển cho một phần cứng mới được tạo ra, trong quá trình này có thể xác định được tỷ lệ lỗi phần cứng cao. Các khiếm khuyết tìm thấy chỉ là không giới hạn ở phần mềm. Nó cũng có thể liên quan đến phần cứng.

#### Cập nhật phần mềm liên tục

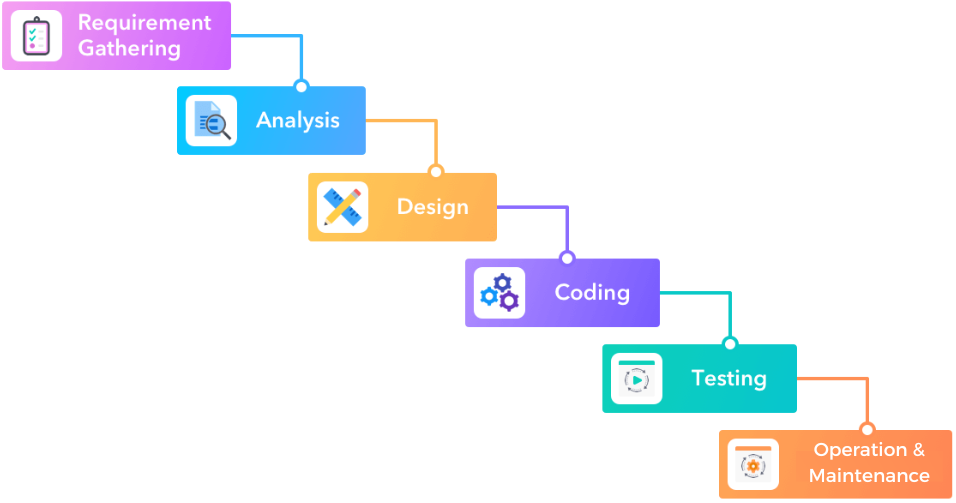
Các hệ thống nhúng yêu cầu cập nhật phần mềm thường xuyên như nâng cấp kernel, sửa lỗi bảo mật, trình điều khiển thiết bị khác nhau, v.v ... Các ràng buộc được xác định với ảnh hưởng cập nhật phần mềm khiến việc nhận dạng lỗi trở nên khó khăn. Ngoài ra, nó làm tăng tầm quan trọng của quy trình xây dựng và triển khai.

## CÁC MÔ HÌNH KIỂM THỬ

Có rất nhiều **mô hình kiểm thử phần mềm**, tuy nhiên, một số mô hình được các tester thường xuyên sử dụng gồm: mô hình thác nước, mô hình chữ V, mô hình mẫu, mô hình Agile, mô hình xoắn ốc, mô hình Scrum, mô hình tăng trưởng và mô hình RAD.

### Mô hình thác nước (Waterfall model)

**Mô hình kiểm thử phần mềm Waterfall model** được sử dụng đầu tiên. Đây là mô hình áp dụng tuần tự các giai đoạn phát triển phần mềm. Theo đó, đầu vào của giai đoạn sau chính là đầu ra của giai đoạn trước.



**Hình 1:** **Mô hình thác nước**

Tester chỉ có thể thực hiện giai đoạn sau khi giai đoạn trước đã kết thúc. Đặc biệt, khi muốn thay đổi, kiểm thử viên không thể quay lại giai đoạn trước để xử lý các yêu cầu. Mô hình thác nước thường được áp dụng trong các dự án nhỏ, ngắn hạn, ít thay đổi và không có yêu cầu cụ thể.

**Các giai đoạn của mô hình thác nước bao gồm:**

* ***Requirement Gathering***: Thu thập, phân tích yêu cầu được ghi lại vào tài liệu đặc tả.
* ***Analysis***: Tiến hành phân tích thiết kế và xác định kiến trúc tổng thể của phần mềm.
* ***Coding***: Hệ thống được phát triển theo từng Unit, đồng thời được tích hợp trong giai đoạn tiếp theo. Unit được phát triển và kiểm thử bởi lập trình viên được gọi là Unit Test.
* **Testing**: Giai đoạn này, công việc chủ yếu là kiểm tra, phát hiện và sửa lỗi để phần mềm hoạt động đúng theo tài liệu đặc tả yêu cầu.
* **Operations and Maintenance**: Khi có thay đổi từ khách hàng, người dùng sẽ tiến hành bảo trì hệ thống.

**Ưu điểm:**

* Thân thiện, dễ tiếp cận, sử dụng và quản lý.
* Các giai đoạn phát triển sản phẩm được xác định rõ rang.
* Xác nhận ở từng giai đoạn để kịp thời phát hiện và sửa lỗi.

**Nhược điểm:**

* Ít linh hoạt, hạn chế phạm vi điều chỉnh.
* Khó để quay lại khi đã kết thúc giai đoạn.
* Gặp khó khăn khi đo lường sự phát triển trong mỗi giai đoạn.
* Không phù hợp với những dự án phức tạp, đang diễn ra, có nhiều thay đổi về yêu cầu trong vòng đời phát triển

### Mô hình mẫu

Mô hình mẫu được phát triển dựa trên những yêu cầu của hệ thống. Khi đó, khách hàng sẽ có cái nhìn tổng quan về hệ thống thực tế. Đây là mô hình kiểm thử phù hợp với những dự án lớn, phức tạp, không thể xác định rõ ràng yêu cầu.

Trong mô hình mẫu, thu thập yêu cầu là việc đầu tiên với sự có mặt của khách hàng và phía phát triển phần mềm để xác định mục tiêu tổng quát của hệ thống phần mềm sau này. Thêm nữa, tất cả yêu cầu được ghi nhận và sơ lược những nhóm yêu cầu cần được làm rõ.

Tiếp theo là thiết kế, tập trung chuyển tải những khía cạnh thông qua Prototype để khách hàng có thể hình dung và đánh giá về hệ thống phần mềm. Đây là việc giúp cho đội ngũ phát triển phần mềm có thể hiểu rõ hơn về những gì cần được phát triển, sau đó tiến hành tinh chỉnh yêu cầu sao cho phù hợp.

Các Prototype thường được làm trong thời gian ngắn cho nên không được thiết kế trên công cụ phát triển và môi trường của giai đoạn xây dựng phần mềm sau này. Prototype không đặt ra mục tiêu tái sử dụng cho giai đoạn phát triển phần mềm thực sự.

**Ưu điểm:**

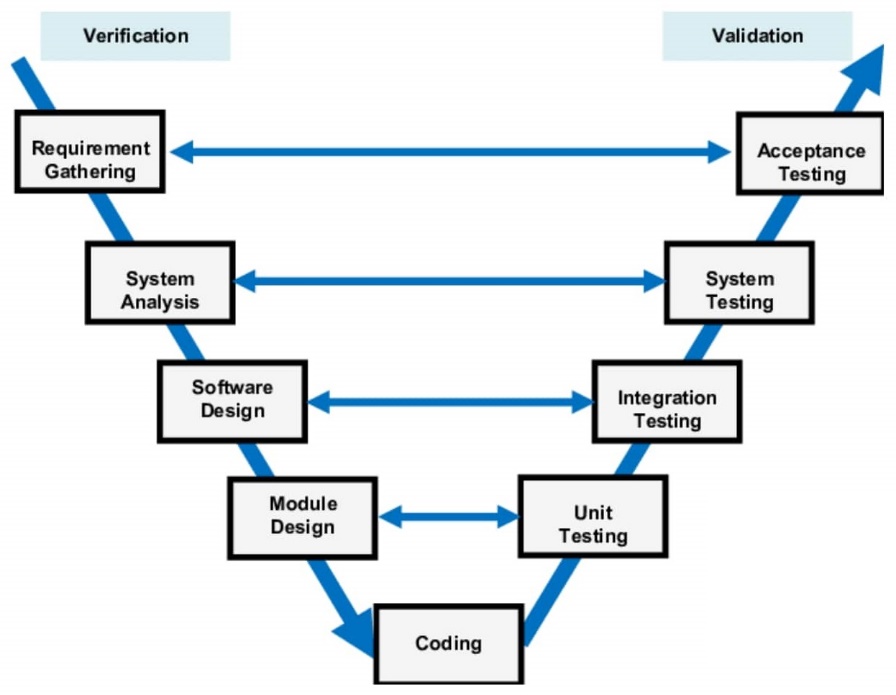
* Cải thiện khoảng cách giữa nhà phát triển phần mềm và người dùng
* Người dùng có thể sớm hình dung đặc điểm và chức năng của hệ thống phần mềm

**Nhược điểm:**

* Mô hình mẫu vẫn chưa thể cải thiện hoàn toàn khoảng cách giữa yêu cầu và ứng dụng cuối cùng
* Prototype thường được làm vội vàng, có thể không phân tích, đánh giá kỹ lưỡng các khía cạnh liên quan đến hệ thống cuối cùng
* Khi Prototype không chuyển tải hết các đặc điểm, chức năng của hệ thống phần mềm thì người dùng có thể “vỡ mộng” và không còn quan tâm đến hệ thống sẽ được phát triển.

### Mô hình chữ V (V model)

**Mô hình chữ V** là mô hình kiểm thử phần mềm được mở rộng hơn so với mô hình thác nước. Mô hình chữ V (V model) dựa trên sự kết hợp của một giai đoạn thử nghiệm cho mỗi giai đoạn phát triển tương ứng.



**Hình 2:** **Mô hình chữ V**

Tức là, mỗi giai đoạn của chu kỳ phát triển phần mềm sẽ có một giai đoạn thử nghiệm liên quan trực tiếp. Mô hình kiểm thử phần mềm chữ V có tính kỷ luật cao, chỉ khi hoàn thành giai đoạn trước mới có thể bắt đầu giai đoạn tiếp theo.

Với mô hình chữ V, việc kiểm thử xuất hiện ngay từ đầu. Từ lúc thu thập yêu cầu là có thể review tài liệu, review code, review đặc tả chi tiết các bản thiết kế, cuối cùng sẽ là test ở mức thấp nhất: từng module, màn hình, chức năng, test tích hợp và kiểm thử hệ thống.

Mô hình chữ V và mô hình thác nước được ứng dụng gần giống nhau, bởi vì chúng đều thuộc loại tuần tự, yêu cầu rõ ràng trước khi bắt đầu dự án. Mô hình chữ V (V model) thường được ứng dụng trong phát triển Y tế.

**Ưu điểm:**

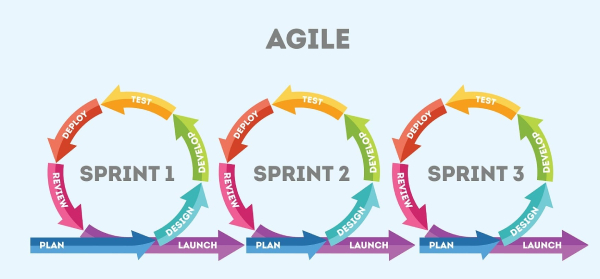
* Đơn giản, dễ hiểu, dễ sử dụng và quản lý.
* Phân phối rõ từng giai đoạn, quy trình đánh giá cụ thể.
* Mô hình có tính kỷ luật cao, các giai đoạn được hoàn thành cùng lúc.
* Phù hợp với những dự án nhỏ và các yêu cầu được hiểu một cách rõ rang.

**Nhược điểm:**

* Khó khăn trong việc quản lý, kiểm soát rủi ro.
* Khó để quay lại và thay đổi chức năng khi ứng dụng đang ở giai đoạn thử nghiệm.
* Mô hình này không phù hợp với những dự án lớn, phức tạp, đang diễn ra và hướng đối tượng.

### ****Mô hình Agile****

Mô hình kiểm thử phần mềm Agile khá linh hoạt, giúp đưa sản phẩm đến tay người dùng càng nhanh càng tốt. Mô hình này có sự cải tiến so với một số mô hình kiểm thử cũ, điển hình là mô hình thác nước (Waterfall model).



**Hình 3:** **Mô hình** Agile

Mô hình Agile gồm hàng loạt phương thức phát triển lặp và tăng dần, trong đó, giải pháp và các yêu cầu được phát triển thông qua sự liên kết giữa các nhóm liên chức năng, nhóm tự quản.

Mô hình Agile dựa trên mô hình Iterative and incremental. Các giải pháp phát triển và yêu cầu dựa trên sự kết hợp của các function. Bên cạnh đó, những tính năng cụ thể của bản phát hành cuối sẽ được cung cấp bởi các tác vụ chia thành các khung thời gian nhỏ.

**Ưu điểm:**

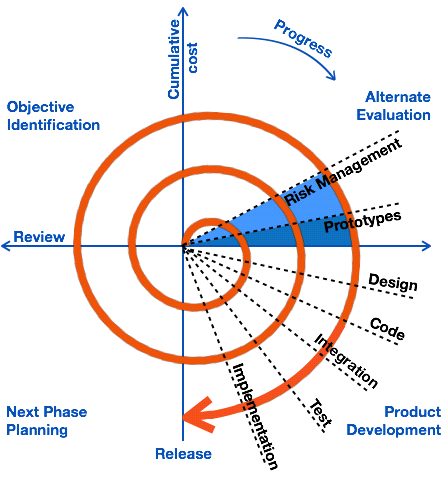
* Dễ hiểu, dễ sử dụng.
* Khách hàng hài lòng vì giao nhanh chóng.
* Dễ dàng thay đổi và bổ sung theo yêu cầu.
* Các chức năng được xây dựng rõ ràng, dễ quản lý.
* Khơi dậy tinh thần làm việc nhóm và trao đổi công việc hiệu quả.
* Nhà phát triển, người thử nghiệm và khách hàng thường xuyên trao đổi.

**Nhược điểm:**

* Cần một nhóm dày dặn kinh nghiệm.
* Cần sự tương tác rõ ràng của khách hang.
* Không phù hợp để xử lý các phụ thuộc phức tạp.
* Xuất hiện nhiều rủi ro về tính bền vững, khả năng mở rộng và bảo trì.
* Gặp khó khăn khi chuyển giao công nghệ cho các thành viên mới trong nhóm.

### ****Mô hình xoắn ốc (Spiral model)****

Xoắn ốc là mô hình kiểm thử phần mềm được phát triển dựa trên sự kết hợp giữa mô hình thác nước (Waterfall model) và mô hình mẫu (Prototype model). Thông thường, mô hình xoắn ốc được dùng cho các ứng dụng lớn hay các hệ thống được xây dựng các phân đoạn/giai đoạn nhỏ.



**Hình 4:** **Mô hình** xoắn ốc

Các pha trong quy trình phát triển mô hình xoắn ốc (Spiral model) bao gồm: lập kế hoạch => đánh giá, giảm thiểu rủi ro => phát triển sản phẩm => đánh giá và lên kế hoạch pha tiếp theo.

Lập kế hoạch (Planning phase): Thu thập và phân tích yêu cầu nhận được từ khách hàng. Cần xác định rõ ràng mục tiêu và đối tượng của từng pha.

Đánh giá, giảm thiểu rủi ro (Alternate evaluation): Xác định, đánh giá và thực hiện những hành động để giảm thiểu rủi ro. Nếu xuất hiện rủi ro trong quá trình này sẽ có các giải pháp thay thế phù hợp. Phát triển sản phẩm (Product development): Lựa chọn mô hình thích hợp để phát triển hệ thống phần mềm.

Đánh giá và lập kế hoạch (Next phase planning): Tiến hành đánh giá dự án và lập kế hoạch cho pha tiếp theo.

**Ưu điểm:**

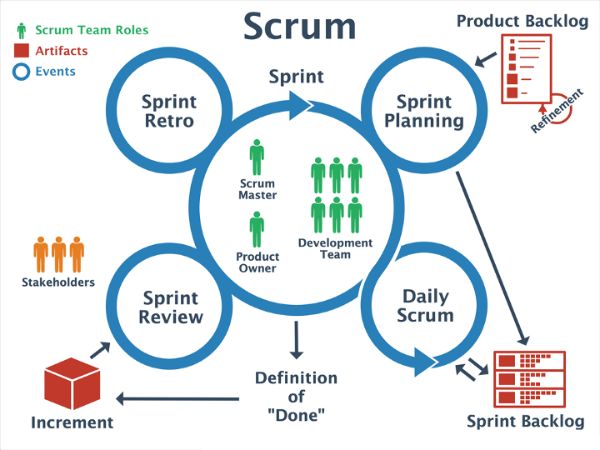
* Dễ dàng kiểm soát rủi ro.
* Hiệu quả đối với phần mềm quy mô lớn.
* Phát hiện sớm những vấn đề quan trọng.

**Nhược điểm:**

* Mô hình này chưa thực sự phổ biến.
* Cần thay đổi thường xuyên dẫn đến lặp vô hạn.
* Phức tạp, không phù hợp với những dự án nhỏ, ít rủi ro.
* Tiêu tốn nhiều thời gian và chi phí cao để hoàn thành dự án.
* Manager cần có kỹ năng tốt để đánh giá rủi ro và quản lý dự án hiệu quả.

### Mô hình Scrum

Mô hình Scrum sử dụng các tiếp cận lặp, tăng trưởng để tối ưu hóa tính khả đoán và kiểm soát rủi ro. Những khía cạnh quan trọng của tiến trình cần được hiển thị cụ thể, rõ ràng cho những người có trách nhiệm trong tiến trình đó.



**Hình 5:** **Mô hình** Scrum

Mô hình Scrum chia các yêu cầu thành từng giai đoạn, mỗi giai đoạn gồm một số lượng nhất định yêu cầu và thường kéo dài từ 1 – 4 tuần (không quá 1 tháng). Đầu mỗi giai đoạn sẽ làm rõ những yêu cầu được thực hiện. Việc tiếp theo là code và test.

Cuối giai đoạn là một sản phẩm hoàn thành cả code và test, có thể demo và chạy được. Khi hoàn thành giai đoạn 1 sẽ bắt đầu giai đoạn 2, giai đoạn 3, giai đoạn 4,… cho đến khi hoàn thành hết các yêu cầu đặt ra.

Ba nhân tố quan trọng cấu thành nên Scrum gồm: tổ chức (Organization), tài liệu (Artifacts), quy trình (Process). Trong đó:

* Tổ chức: Tổ chức nhóm và roles, người sở hữu sản phẩm, người điều phối, nhóm phát triển.
* Tài liệu: Danh sách chức năng cần phát triển của sản phẩm, chức năng cần phát triển cho từng giai đoạn, kết quả ước lượng của nhóm.
* Quy trình: Hoạch định, tổng kết cho mỗi giai đoạn và nhận xét theo ngày.

**Ưu điểm:**

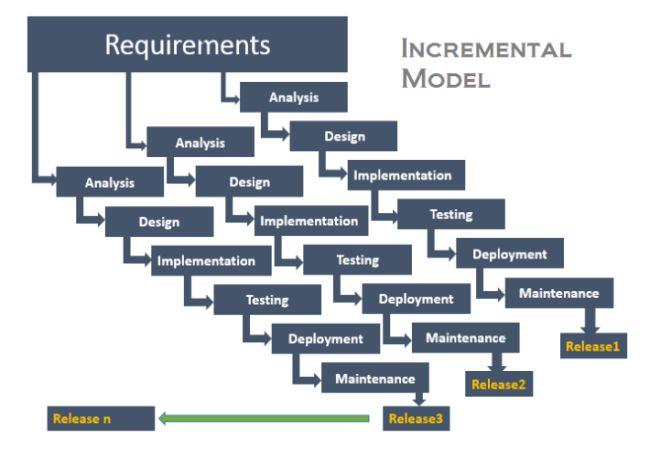
* Có khả năng phát hiện lỗi sớm.
* Một người có thể đảm nhận nhiều việc.
* Phù hợp với dự án mà khách hàng không yêu cầu rõ ràng từ đầu.

**Nhược điểm:**

* Yêu cầu nhóm có kỹ năng nhất định.
* Nhóm phải có sự hiểu biết về mô hình Agile.
* Xác định thời gian và ngân sách gặp nhiều khó khan.
* Kéo dài thời gian vì phải thực hiện theo yêu cầu của khách hang.
* PO có vai trò quan trọng, nếu không làm tốt sẽ ảnh hưởng đến kết quả chung.

### Mô hình tăng trưởng

Mô hình tăng trưởng sẽ chia chu kỳ phát triển phần mềm thành các mô-đun nhỏ để dễ dàng quản lý. Đây là mô hình kiểm thử phần mềm mà mỗi mô-đun sẽ trải qua rất nhiều giai đoạn: phân tích yêu cầu, thiết kế, thực hiện và thử nghiệm, giống như vòng đời phát triển thông thường.



**Hình 5:** **Mô hình** tăng trưởng

Mô hình tăng trưởng phù hợp với những dự án mà yêu cầu đã được mô tả rõ ràng và khách hàng có nhu cầu về sản phẩm sớm. Cũng giống như những mô hình kiểm thử phần mềm khác, mô hình tăng trưởng sở hữu nhiều ưu điểm nổi bật, tuy nhiên nó vẫn tồn tại một số hạn chế nhất định.

**Ưu điểm:**

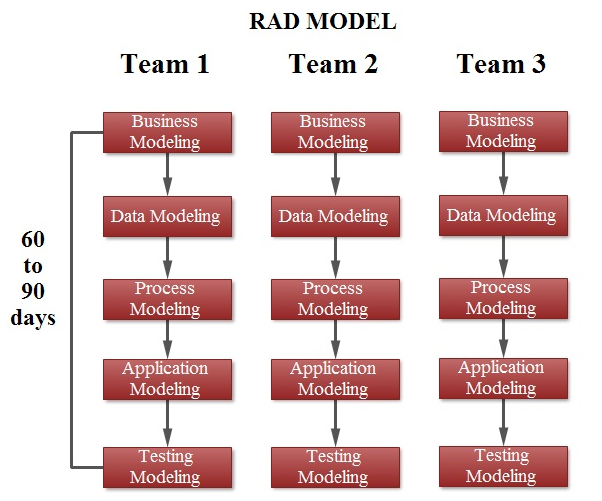
* Phát triển dễ dàng, nhanh chóng
* Kiểm tra và sửa lỗi đơn giản hơn
* Linh hoạt, ít tốn kém khi thay đổi yêu cầu, phạm vi

**Nhược điểm:**

* Yêu cầu cao về lập plan và thiết kế
* Tổng chi phí của mô hình này cao hơn so với mô hình thác nước

### Mô hình RAD

Mô hình kiểm thử phần mềm RAD (Rapid Application Development) sử dụng quy hoạch tối thiểu, có lợi cho việc tạo mẫu nhanh. Để tạo ra và phân phối sản phẩm nhanh hơn, các mô-đun chức năng được phát triển song song và tích hợp.



**Hình 3:** **Mô hình** RAD

Khi áp dụng mô hình RAD, dự án có thể không thành công nếu nó không được chia thành các mô-đun. Nên sử dụng mô hình RAD khi có nhu cầu tạo hệ thống mà yêu cầu của khách hàng thay đổi trong khoảng thời gian ngắn (2 – 3 tháng), chi phí cao và có sẵn designer cho model.

**Ưu điểm:**

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

**Nhược điểm:**

* Nhóm thực hiện cần có kỹ năng nhất định.
* Mô hình này chỉ phù hợp với những hệ thống có mô-đun.

## LỰA CHỌN MÔ HÌNH PHÁT TRIỂN PHẦN MỀM

* Bằng việc phân tích và so sánh các mô hình phát triển phần mềm theo những tiêu chí ở trên thì có thể thấy mỗi loại mô hình đều có ưu, nhược điểm riêng.
* Mỗi mô hình sẽ được áp dụng với từng dự án cụ thể và phù hợp nhất. Trong đó mô hình đi sau sẽ là sự cải tiến và phát triển hơn so với mô hình đi trước.
* Sự thay đổi này giúp các nhà quản lý có sự lựa chọn đa dạng về sản phẩm, dự án. Tuy nhiên cần chọn mô hình thích hợp với tiến trình phát triển của phần mềm cùng yêu cầu của khách hàng.
* Với những dự án nhỏ, kinh phí thấp cùng với các yêu cầu điều kiện rõ ràng từ đầu thì bạn nên lựa chọn mô hình thác nước. Còn nếu khách hàng muốn sử dụng các tính năng sản phẩm theo giai đoạn hoặc yêu cầu chưa rõ thì nhà quản lý có thể lựa chọn mô hình Scrum hoặc mô hình khác.