Aptech™
COMPUTER EDUCATION
Unleash your potential

SESSION 02

Vài trò của Kiểm tra trong vòng đời phần mềm

Nội dung

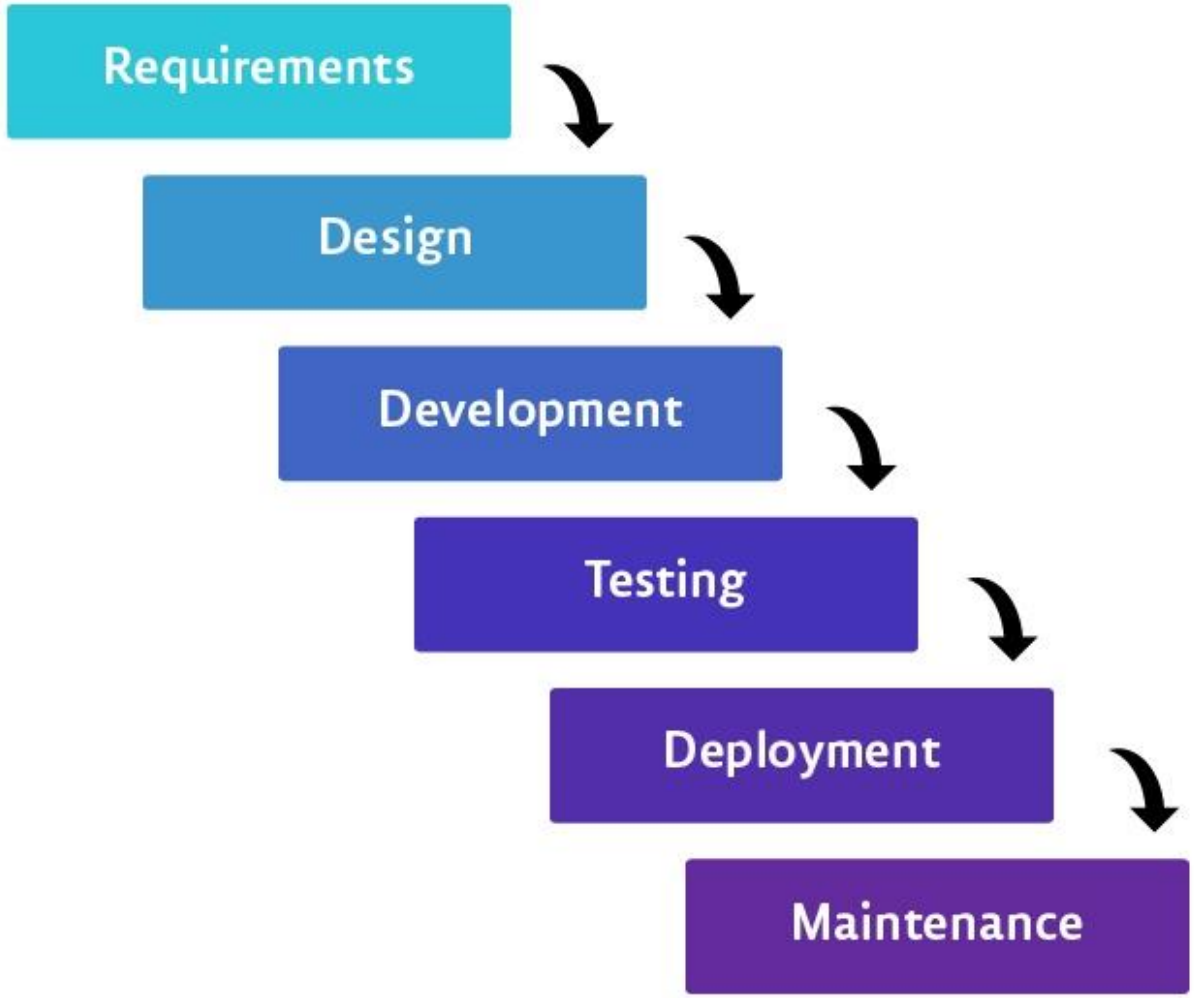
- Những mô hình phát triển phần mềm
  - Mô hình chữ V
  - Mô hình phát triển lặp gia tăng
- Kiểm tra trong mô hình vòng đời phần mềm
- Kiểm chứng và chứng thực (V & V)
- Các mức kiểm tra
- Các kiểu kiểm tra

Những mô hình phát triển phần mềm

- Quan hệ giữa phát triển và kiểm tra
- Thích hợp giữa mô hình đến ngữ cảnh
- Lý do của các mức kiểm thử khác nhau

Mô hình thác nước là gì?

- Mô hình Waterfall hay còn gọi là mô hình thác nước. Được biết tới là một trong những mô hình quản lý dự án dễ hiểu nhất hiện nay, mô hình Waterfall là một phương pháp quản lý dự án dựa trên quy trình thiết kế tuần tự và liên tiếp.
- Trong mô hình Waterfall, các giai đoạn của dự án được thực hiện lần lượt và nối tiếp nhau. Giai đoạn mới chỉ được bắt đầu khi giai đoạn trước nó đã được hoàn thành.



Các giai đoạn của mô hình Waterfall

- Yêu cầu
- Thiết kế
- Thực hiện (xây dựng)
- Kiểm chứng
- Triển khai
- Bảo trì

Mô hình thác nước là gì?

GIAI ĐOẠN YÊU CẦU (REQUIREMENT ANALYSIS)

- Nhóm thực hiện tìm kiếm các yêu cầu liên quan đến dự án.
- Ví dụ:
  - Xác định dự án sẽ giải quyết nhu cầu kinh doanh nào
  - Yêu cầu của người dùng đối với sản phẩm được phát triển bởi dự án
  - Các ràng buộc và rủi ro đi kèm.

Mô hình thác nước là gì?

GIAI ĐOẠN THIẾT KẾ (DESIGN)

- Nhóm tạo ra thiết kế cho sản phẩm để giải quyết mọi yêu cầu, ràng buộc và mục tiêu thiết kế.
- Một bản thiết kế điển hình sẽ được hoàn thành một cách càng cụ thể càng tốt. Nó sẽ mô tả chính xác logic của hệ thống được đề cập trong phần phân tích sẽ được thực thi như nào.

Mô hình thác nước là gì?

GIAI ĐOẠN XÂY DỰNG (DEVELOPMENT)

- Sản phẩm được chế tạo để hỗ trợ thiết kế.
- Đôi khi, sản phẩm được xây dựng trong các đơn vị dùng để thí nghiệm và tích hợp trong giai đoạn tiếp theo.

Mô hình thác nước là gì?

GIAI ĐOẠN KIỂM CHỨNG (TEST)

- Các bộ phận của sản phẩm được kiểm tra.
- Nếu cần sẽ được tích hợp lại với nhau để thử nghiệm.
- Toàn bộ hệ thống được kiểm tra để tìm ra lỗi và đảm bảo các mục tiêu thiết kế.

Mô hình thác nước là gì?

GIAI ĐOẠN TRIỂN KHAI (DEPLOYMENT)

- Sản phẩm được thử nghiệm thực sự đi vào hoạt động.
- Đối với các dự án thuộc lĩnh vực công nghệ thông tin, sản phẩm được triển khai vào môi trường để người dùng có thể bắt đầu sử dụng nó.
- Đối với một dự án xây dựng, giai đoạn triển khai là khi tòa nhà hoàn toàn sẵn sàng cho người ở.

Mô hình thác nước là gì?

GIAI ĐOẠN BẢO TRÌ (MAINTENANCE)

- Là một khoảng thời gian giám sát ngắn. Trong đó nhóm dự án giải quyết các vấn đề của khách hàng.
- Đối với các dự án phần mềm, điều này thường có nghĩa phát hành các bản vá và cập nhật để sửa vấn đề.
- Trong các dự án khác, các điều chỉnh về môi trường được thực hiện để giải quyết vấn đề. Chẳng hạn như tối ưu hóa điều hòa không khí trong một tòa nhà mới.

Mô hình thác nước là gì?

ƯU ĐIỂM CỦA MÔ HÌNH WATERFALL

- Dù mô hình thác nước đã dần dần biến mất trong vài năm trở lại đây nhường chỗ cho các mô hình linh hoạt (Agile) hơn. Nhưng nó vẫn đem lại một số lợi ích, đặc biệt trong các dự án và tổ chức lớn mà cần các giai đoạn và hạn hoàn thành của công việc.

Mô hình thác nước là gì?

ƯU ĐIỂM CỦA MÔ HÌNH WATERFALL

- Thích nghi tốt với những nhóm linh hoạt
- Áp đặt một tổ chức có kết cấu chặt chẽ
- Cho phép những thay đổi thiết kế sớm
- Thích hợp cho những dự án theo hướng đến mốc

Mô hình thác nước là gì?

NHƯỢC ĐIỂM CỦA MÔ HÌNH WATERFALL

- Nó không phải là một mô hình lý tưởng cho một dự án kích thước lớn.
- Nếu yêu cầu không rõ ràng ngay từ đầu thì đó là phương pháp kém hiệu quả hơn.
- Rất khó di chuyển trở lại cái giai đoạn trước đó để thay đổi.
- Quá trình thử nghiệm bắt đầu khi quá trình phát triển kết thúc. Do đó, nó có nguy cơ cao của các lỗi được tìm thấy sau giai đoạn phát triển, và rất tốn kém để sửa các lỗi.

Mô hình thác nước là gì?

KHI NÀO NÊN ÁP DỤNG MÔ HÌNH WATERFALL?

- Việc áp dụng mô hình Waterfall được khuyến khích khi người thực hiện nắm rõ yêu cầu của dự án tốt nhất, đòi hỏi về tính rõ ràng và tính ổn định cao như:
  - Nắm vững được công nghệ phát triển của công nghệ.
  - Loại bỏ những yêu cầu mập mờ, không rõ ràng.
  - Có lượng tài nguyên phát triển phong phú và trình độ chuyên môn, kỹ thuật cao.
  - Có thể phù hợp cho dự án nhỏ, ngắn hạn.

Mô hình V là gì?

- Một trong những hạn chế lớn nhất của mô hình phát triển phần mềm thác nước là: Các khiếm khuyết được tìm thấy rất chậm trong quá trình phát triển vì kiểm thử được thực hiện vào cuối chu kỳ phát triển.
- Fix bug càng chậm thì càng khó khăn và tốn kém.
- Để khắc phục vấn đề này, một mô hình phát triển mới được giới thiệu gọi là "Mô hình V"

Tham khảo: What is STLC V-Model? (softwaretestinghelp.com)

Mô hình V là gì?

- Mô hình V hiện nay là một trong những quy trình phát triển phần mềm được sử dụng rộng rãi nhất.
- Trong mô hình V việc thực hiện kiểm tra được diễn ra ngay từ giai đoạn lấy yêu cầu.
- V mô hình cũng được gọi là mô hình xác minh (verification) và mô hình xác nhận (validation).

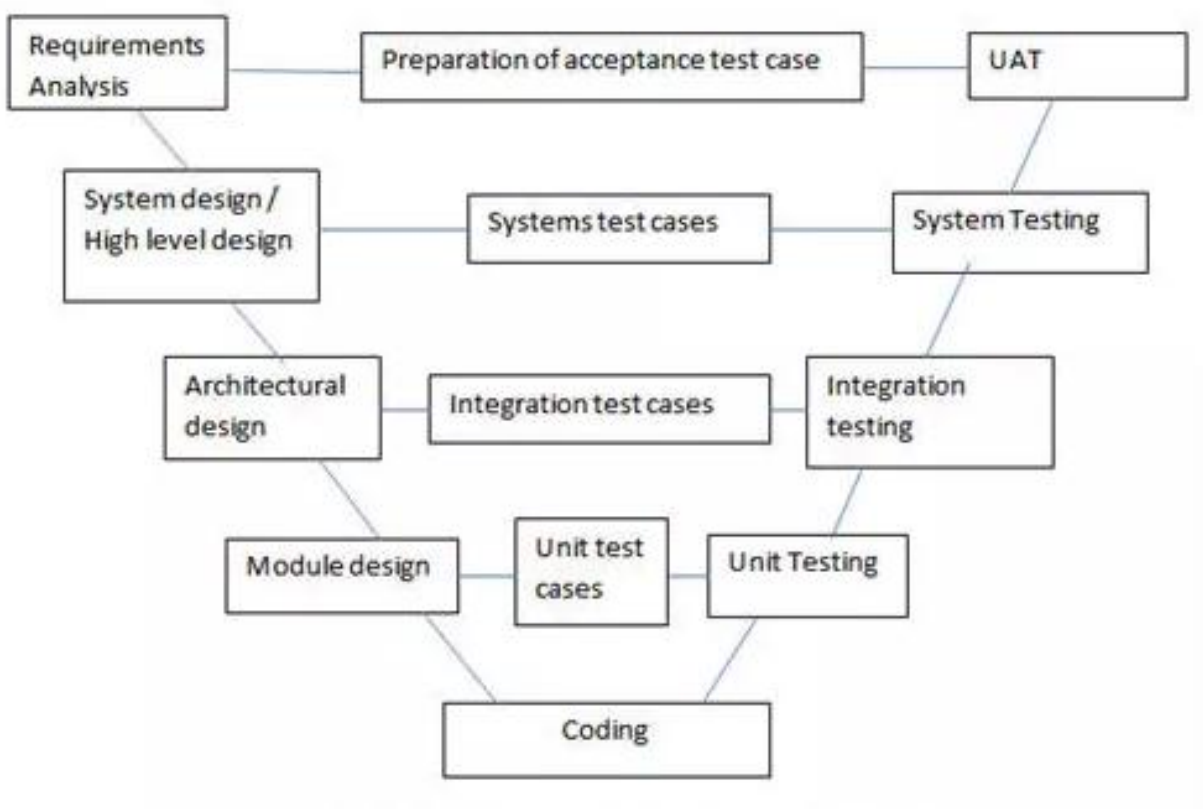
Mô hình V là gì?

- Xác minh (verification): Xác minh là một kỹ thuật phân tích tĩnh. Trong kiểm thử, kỹ thuật này được thực hiện mà không phải chạy code. Nó bao gồm một số hoạt đông như xem lại (review), kiểm tra (inspection) và kiểm tra từ đầu tới cuối (walkthrough).

- Xác nhận (validation): Xác nhận là một kỹ thuật phân tích động, trong đó việc kiểm thử được thực hiện bằng cách thực hiện code. Ví dụ bao gồm kỹ thuật kiểm tra chức năng (function) và phi chức năng (non-function).

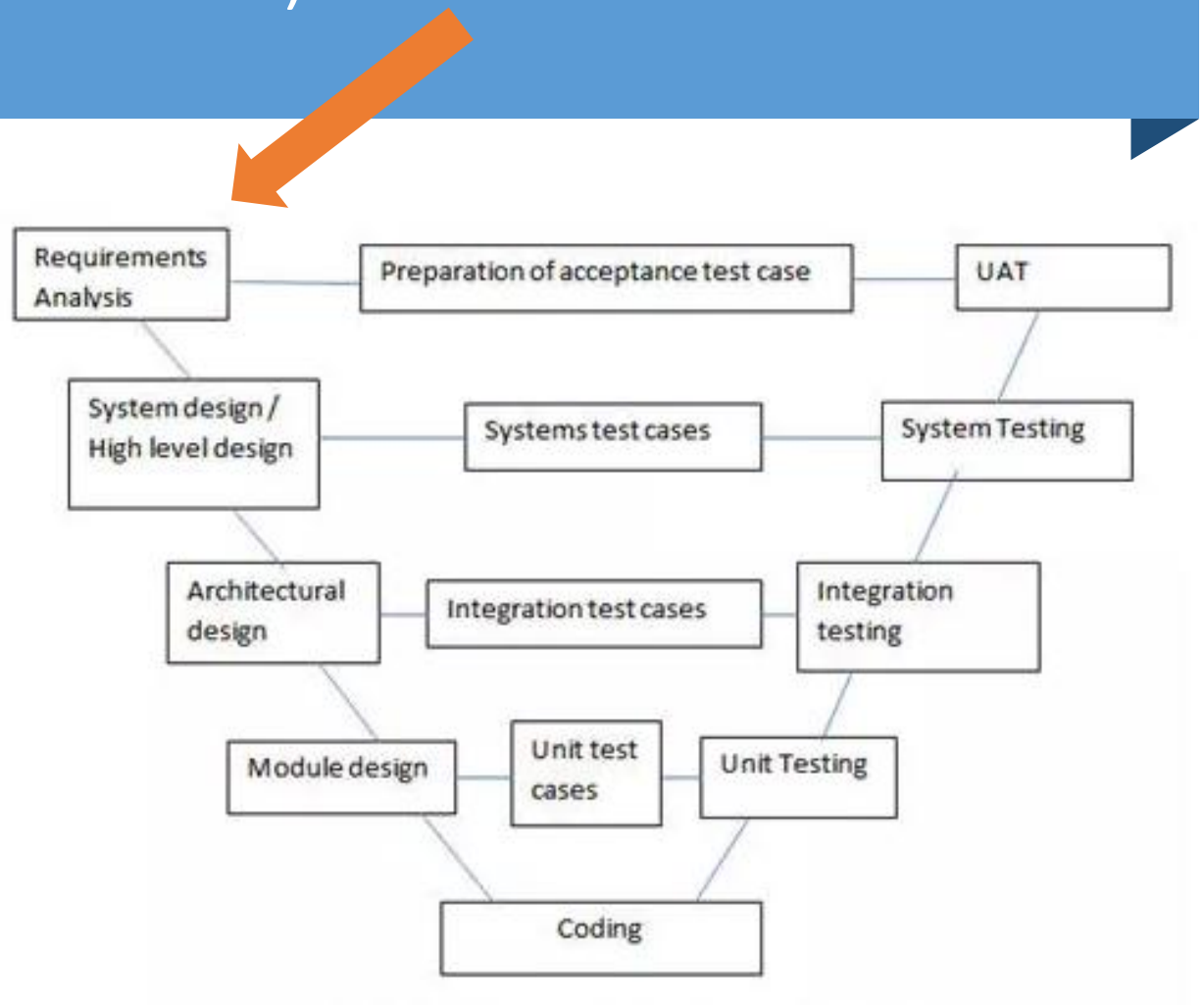
Mô hình V (Sơ đồ)

- Trong mô hình V, các hoạt động phát triển và đảm bảo chất lượng được thực hiện đồng thời.
- Không có pha rời rạc được gọi là kiểm thử, thay vào đó kiểm thử được bắt đầu ngay từ giai đoạn lấy yêu cầu.
- Các hoạt động xác minh và xác nhận đi liền với nhau.



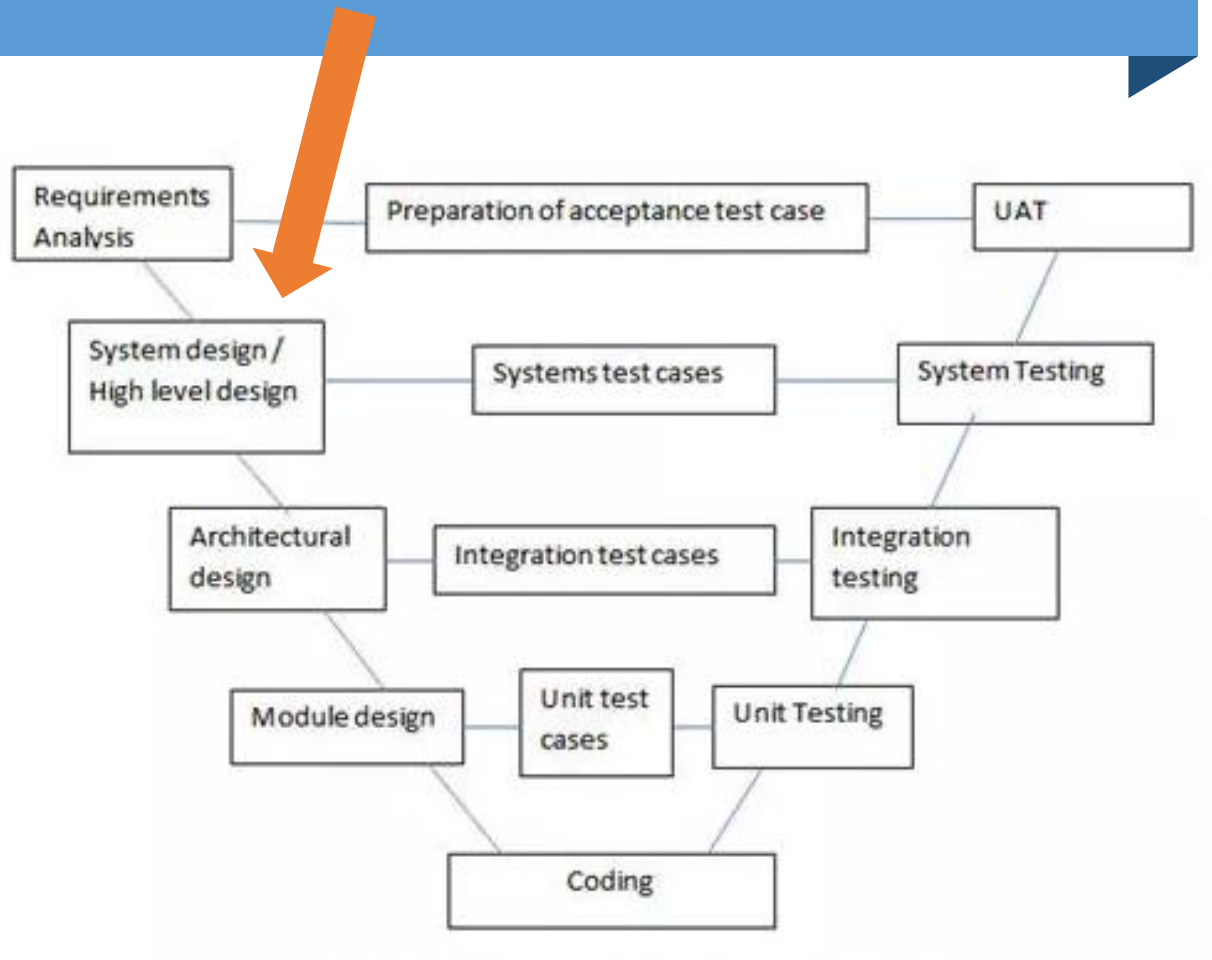
Mô hình V (Phân tích yêu cầu)

- Trong giai đoạn này các yêu cầu được thu thập, phân tích và nghiên cứu. Ở giai đoạn này việc hệ thống chạy như thế nào không quan trọng, quan trọng là hệ thống có những chức năng gì. Brain storming hay walkthrough, interviews cần được thực hiện để có mục tiêu rõ ràng.
- Hoạt động xác minh:Đánh giá yêu cầu (Requirements review).
- Hoạt động xác nhận: Tạo test case UAT (User acceptance test- kiểm thử chấp nhận) = Đầu ra cần có: Tài liệu hiểu về yêu cầu, UAT test case.



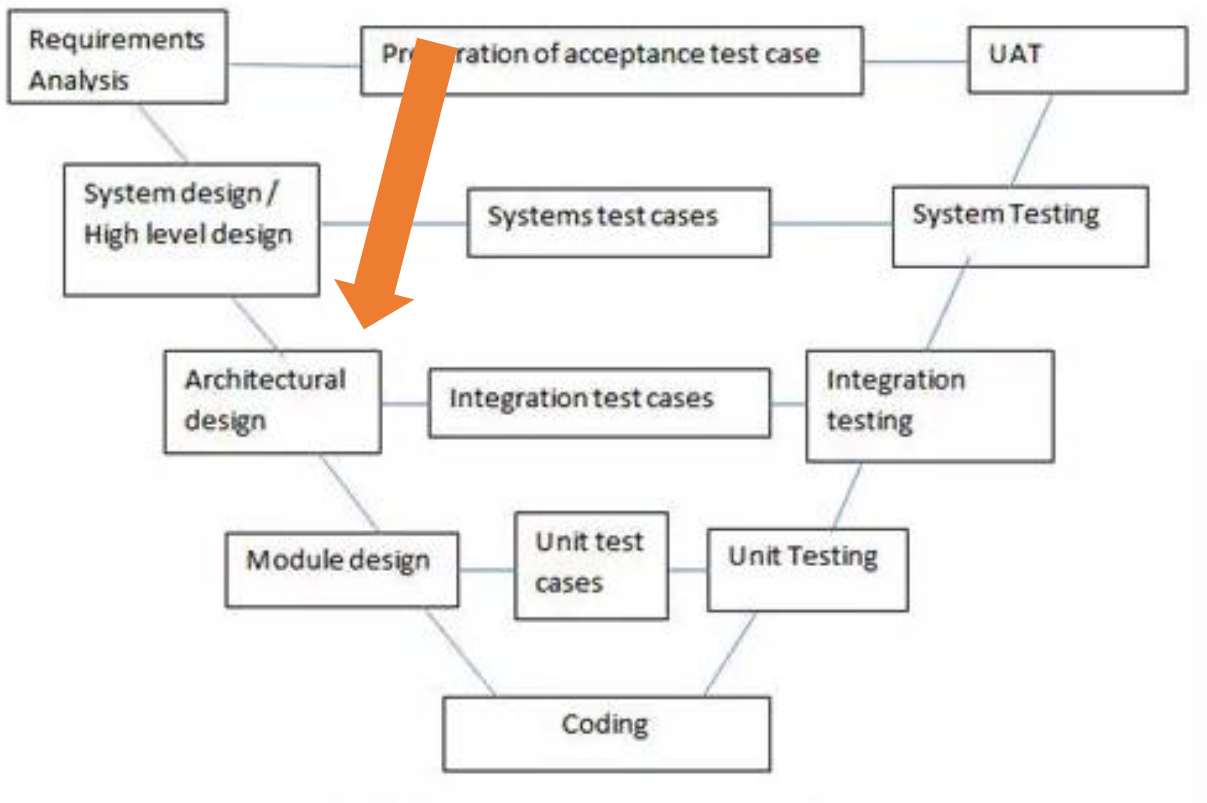
Mô hình V (Yêu cầu hệ thống / High level design)

- Trong giai đoạn này, high level design của phần mềm được xây dựng. Nhóm sẽ nghiên cứu và điều tra về các yêu cầu có thể được thực hiện như thế nào. Tính khả thi về mặt kỹ thuật của yêu cầu cũng được tìm hiểu. Nhóm cũng tìm hiểu về các mô-đun sẽ được tạo / phụ thuộc, nhu cầu phần cứng / phần mềm.
    - Hoạt động xác minh: Đánh giá thiết kế (Design reviews)
    - Hoạt động xác nhận: Tạo test plan và test case, tạo ma trận truy vết (traceability metrics)
    - Đầu ra cần có: System test cases, Feasibility reports, System test plan, tài liệu về yêu cầu phần cứng và các mô đun, ...



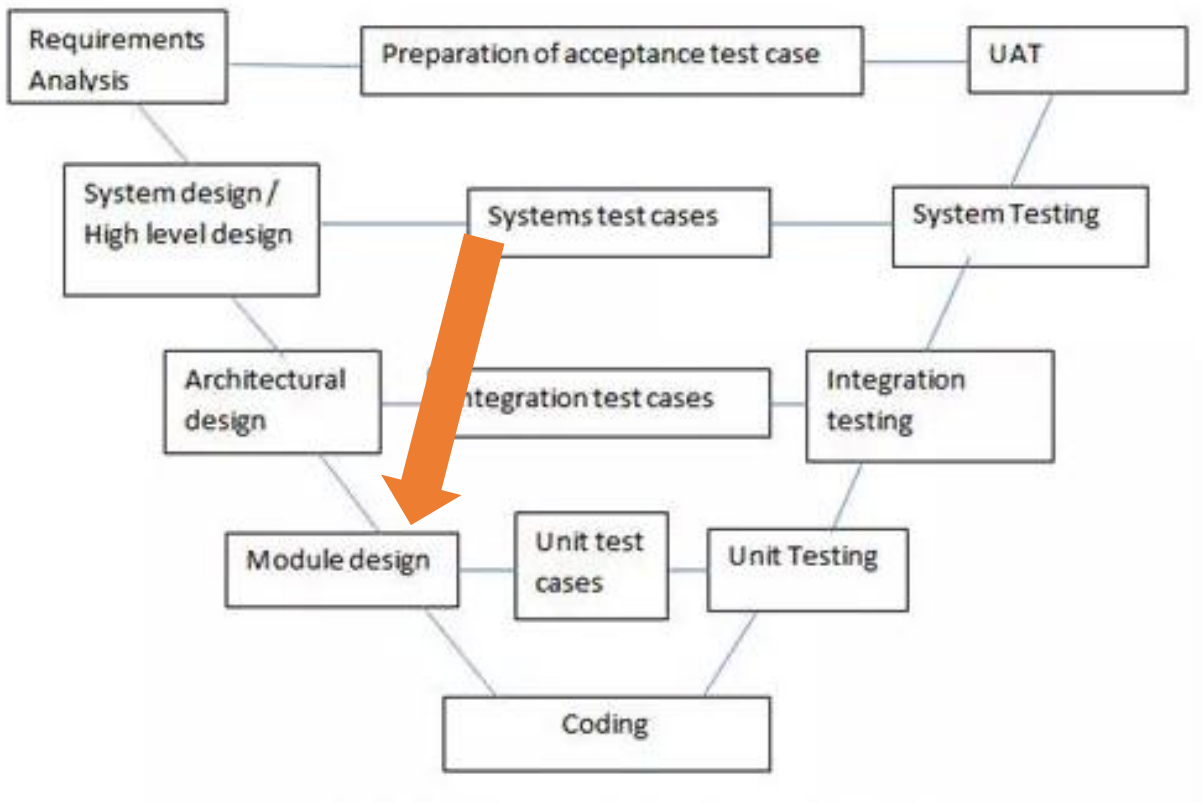
Mô hình V (Thiết kế kiến trúc)

- Trong giai đoạn này, dựa trên thiết kế mức cao, kiến trúc phần mềm được tạo ra. Các mô-đun, mối quan hệ và sự phụ thuộc của họ, sơ đồ kiến trúc, bảng cơ sở dữ liệu, chi tiết về công nghệ đều được hoàn tất trong giai đoạn này.
  - Hoạt động xác minh: Đánh giá thiết kế
  - Hoạt động xác nhận: Kế hoạch thử nghiệm tích hợp và các trường hợp thử nghiệm.
  - Đầu ra cần có: Tài liệu thiết kế, Kế hoạch kiểm thử tích hợp và các trường hợp thử nghiệm, Thiết kế bảng cơ sở dữ liệu,...



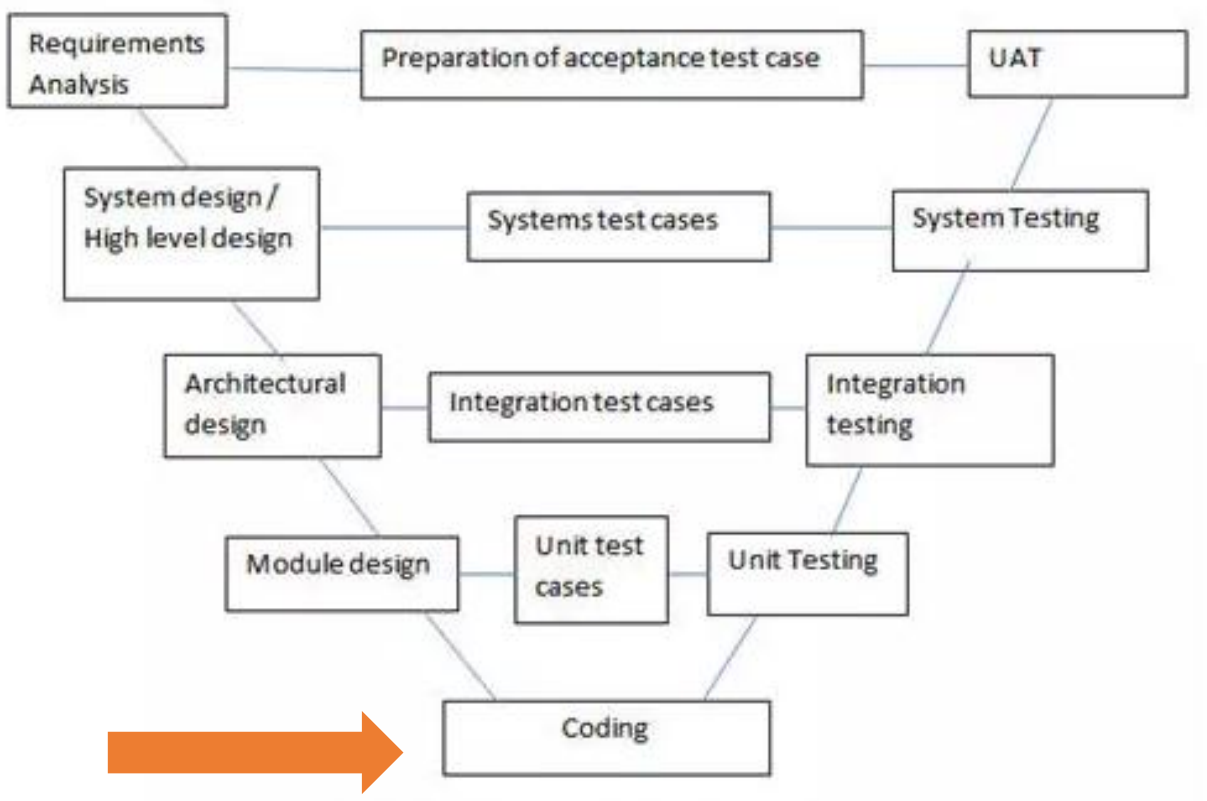
Mô hình V (Thiết kế mô đun / Thiết kế cấp thấp)

- Trong giai đoạn này mỗi mô-đun hoặc các thành phần phần mềm đều được thiết kế riêng. Các method, class, giao diện, các kiểu dữ liệu vv đều được hoàn tất trong giai đoạn này.
  - Hoạt động xác minh: Đánh giá thiết kế
  - Hoạt động xác nhận: Tạo và xem xét các trường hợp kiểm tra đơn vị.
  - Đầu ra cần có: Các đơn vị kiểm tra đơn vị



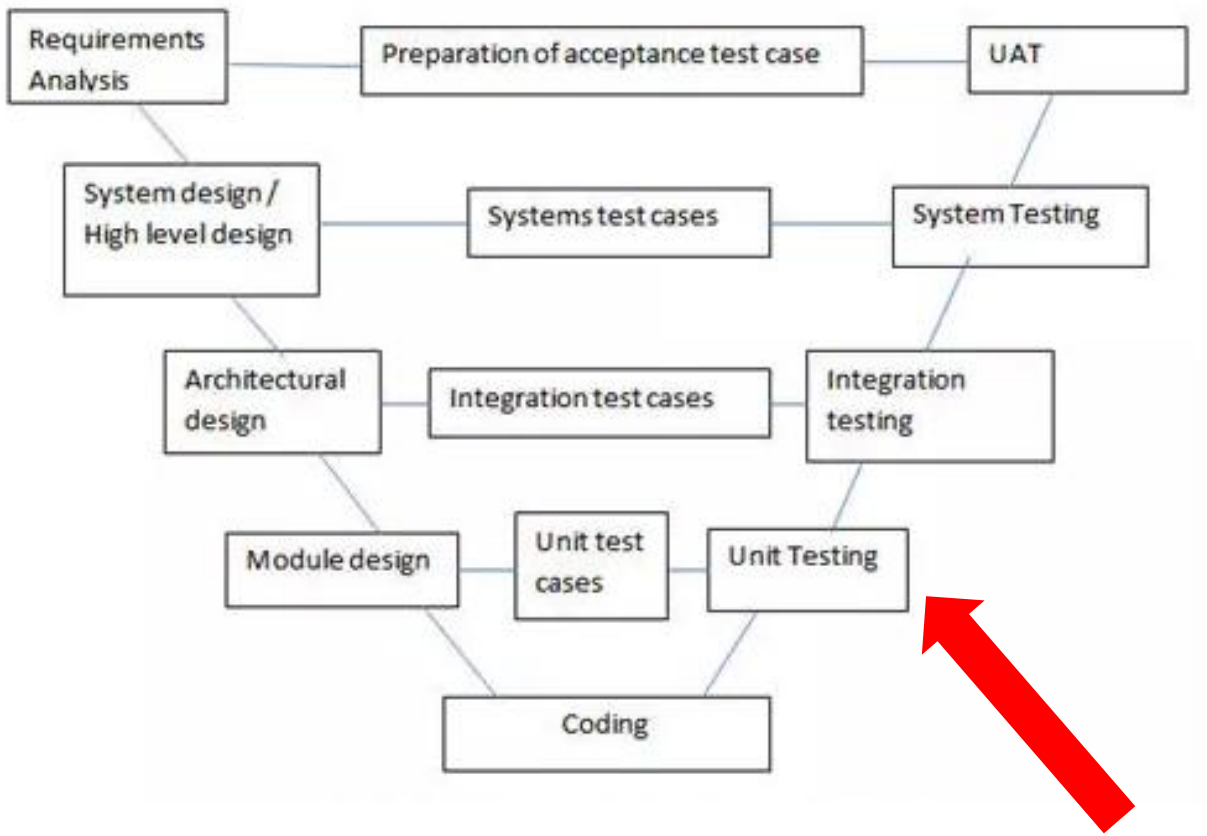
Mô hình V (Implement / Code)

- Trong giai đoạn này, code được thực hiện.
  - Hoạt động xác minh: Xem xét mã, kiểm tra các trường hợp kiểm tra
  - Hoạt động xác nhận: Tạo các trường hợp kiểm tra chức năng.
  - Đầu ra cần có: các trường hợp thử nghiệm, danh sách kiểm tra xem lại.



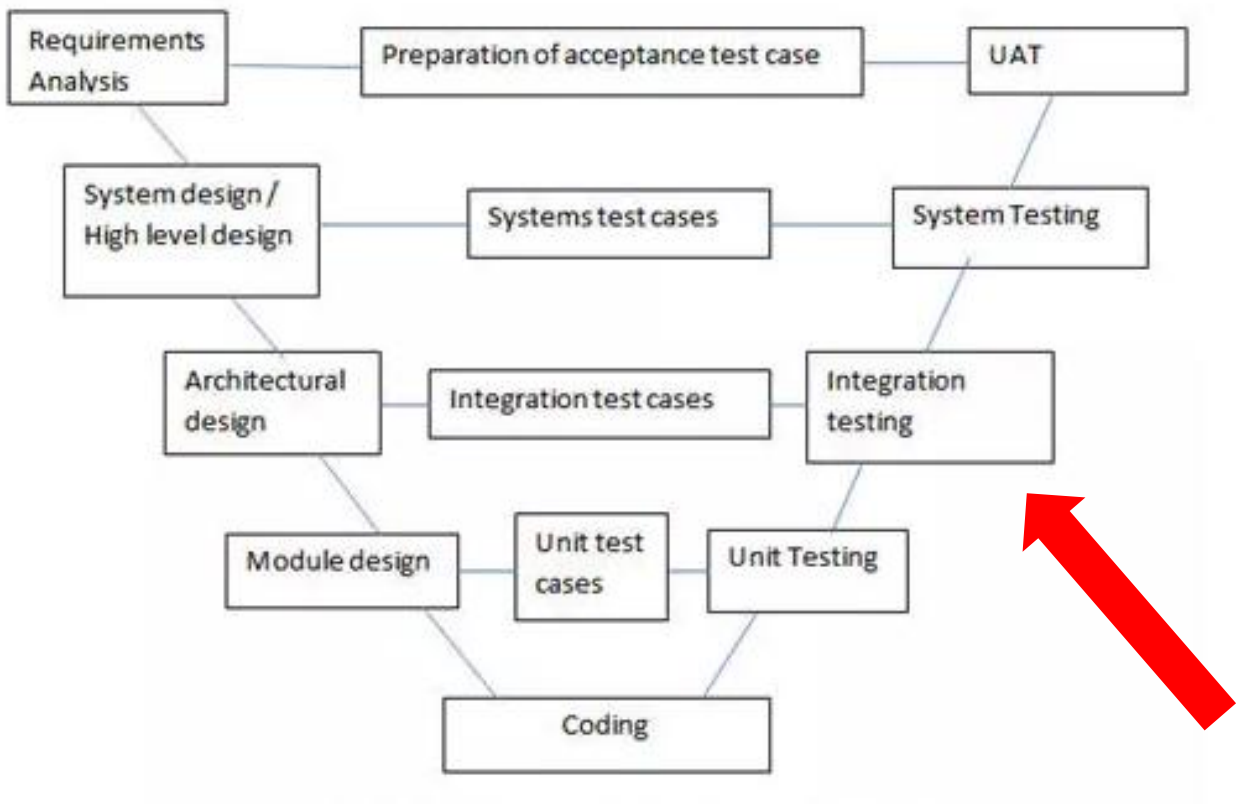
Mô hình V (Kiểm tra đơn vị (unit test))

- Trong giai đoạn này tất cả unit test case đã được tạo ra trong giai đoạn thiết kế cấp thấp sẽ được thực hiện.
  - Unit test là một kỹ thuật kiểm tra hộp trắng, nơi một đoạn code được viết, nó sẽ tích hợp một phương pháp (hoặc một đoạn code khác) để kiểm tra xem đoạn code có cho kết quả mong muốn hay không. Thử nghiệm này về cơ bản được thực hiện bởi nhóm phát triển. Trong trường hợp có bất thường, bug được ghi lại và theo dõi.
  - Đầu ra cần có: Kết quả thực hiện unit test



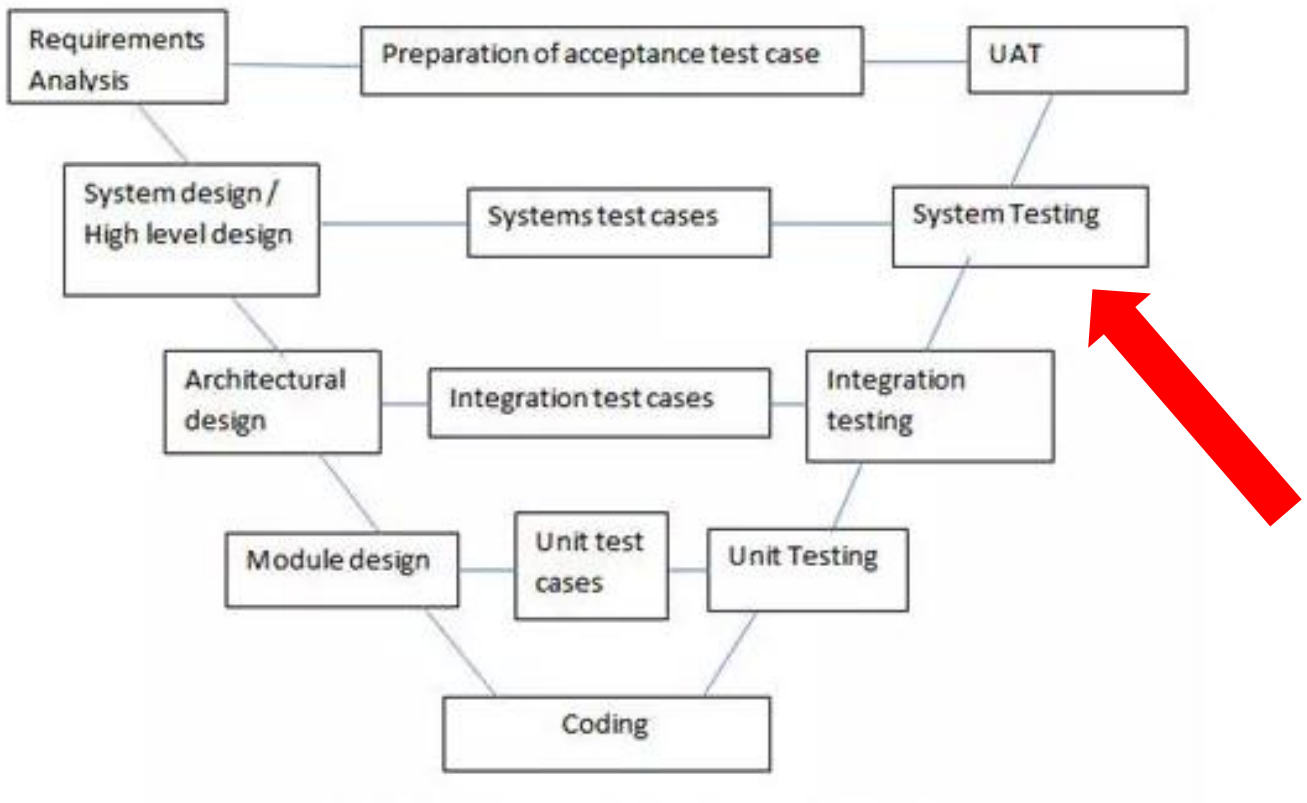
Mô hình V (Kiểm tra tích hợp)

- Trong giai đoạn này các trường hợp kiểm thử tích hợp được thực hiện đã được tạo ra trong giai đoạn thiết kế kiến trúc. Trong trường hợp có bất kỳ dị thường, bug được ghi lại và theo dõi.
  - Kiểm thử tích hợp (Integration Testing): Kiểm thử tích hợp là một kỹ thuật mà đơn vị kiểm tra là mô-đun được tích hợp và kiểm tra xem các mô-đun tích hợp vào có cho kết quả mong đợi không. Nói một cách đơn giản, nó xác nhận xem các thành phần của ứng dụng có hoạt động với nhau như mong đợi hay không.
  - Đầu ra cần có: Các kết quả kiểm tra tích hợp.



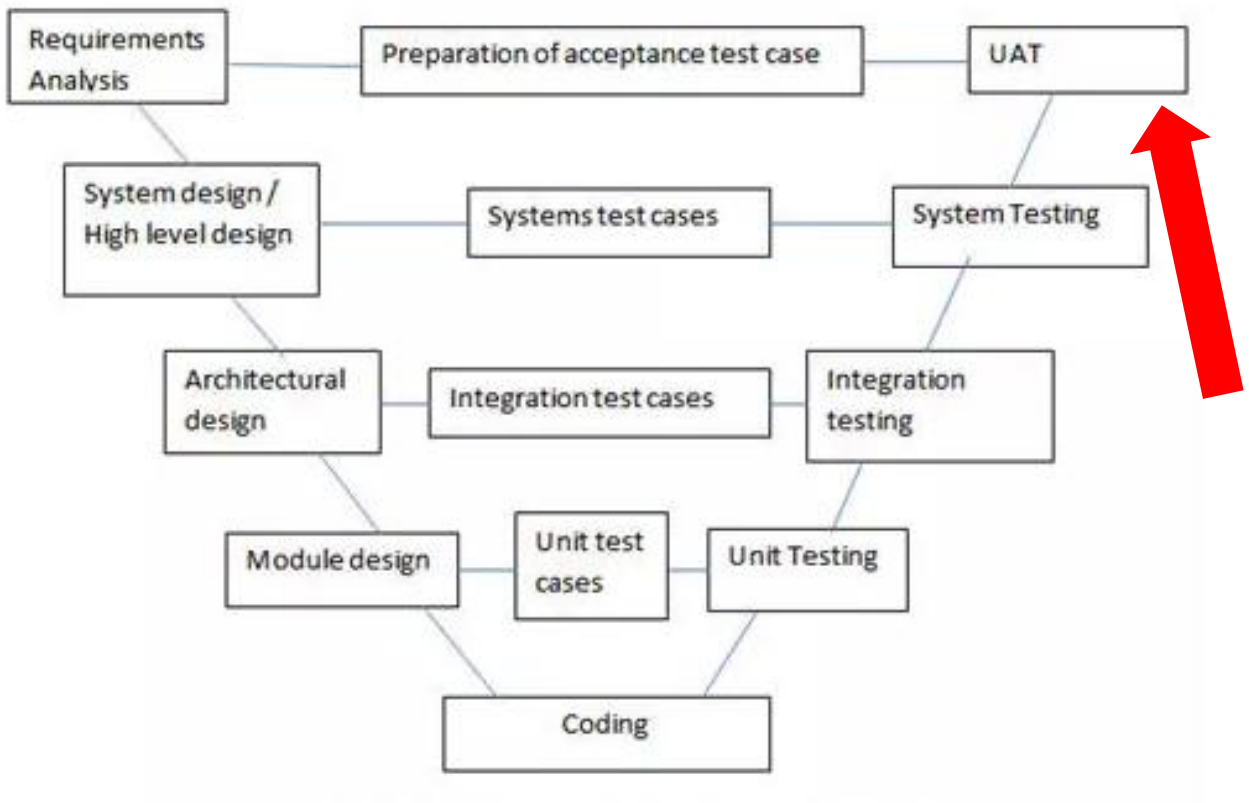
Mô hình V (Kiểm tra hệ thống (System testing))

- Trong giai đoạn này, kiểm thử chức năng và phi chức năng được thực hiện. Nói cách khác, việc kiểm tra thực tế hoạt động của ứng dụng diễn ra ở đây. Lỗi được phát hiện và theo dõi để sửa. Báo cáo tiến độ cũng là một phần quan trọng trong giai đoạn này. Ma trận truy vết (traceability metrics) được cập nhật để kiểm tra mức độ bao phủ và rủi ro được giảm bớt.
  - Đầu ra cần có: Kết quả kiểm tra, Các bản ghi kiểm tra, báo cáo lỗi, báo cáo tóm tắt kiểm tra và các ma trận truy xuất cập nhật.



Mô hình V (Thử nghiệm chấp nhận người dùng)

- Thử nghiệm chấp nhận về cơ bản liên quan đến việc kiểm tra các yêu cầu business. Ở đây kiểm tra được thực hiện để xác nhận rằng các yêu cầu kinh doanh được đáp ứng trong môi trường người dùng. Thử nghiệm khả năng tương thích và đôi khi thử nghiệm phi chức năng (Load, Stress và Volume) cũng được thực hiện trong giai đoạn này.
  - Đầu ra cần có: Kết quả UAT, Ma trận độ bao phủ business được cập nhật.

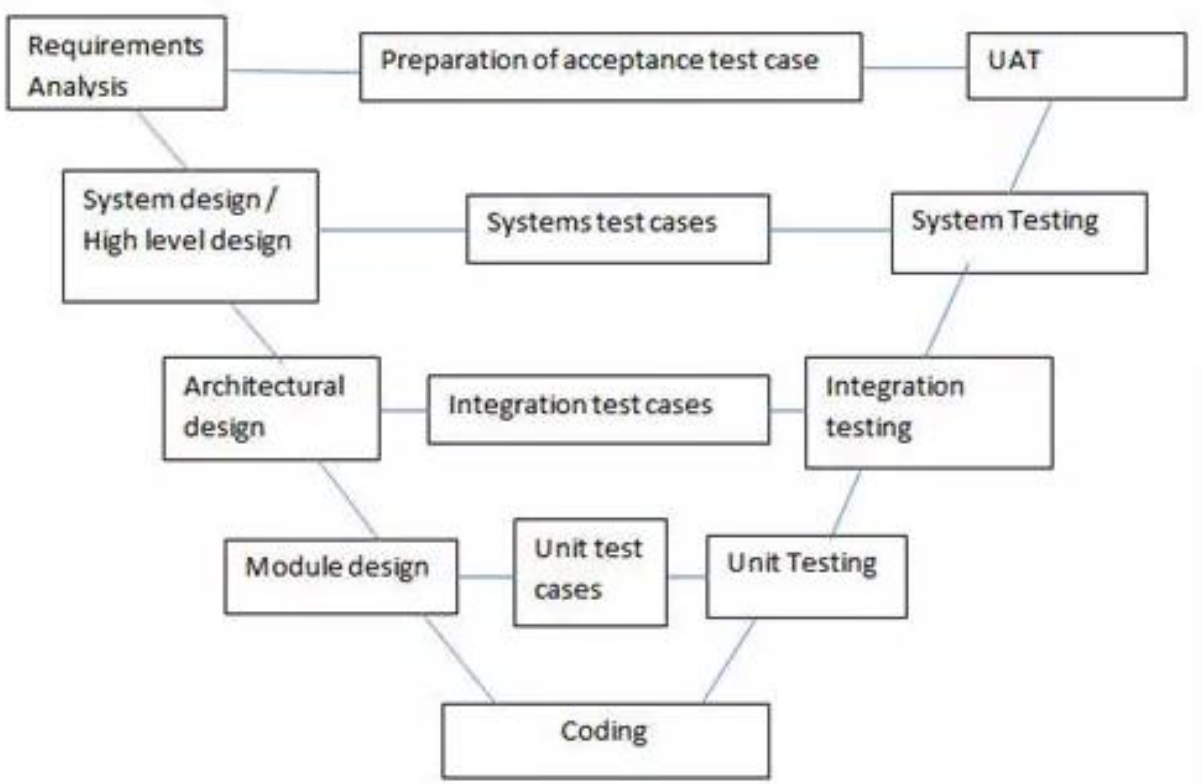


Mô hình V (Khi nào áp dụng)

- Dự án có kích thước nhỏ, và trung bình, các yêu cầu dự án là rõ ràng, cố định.
- Khi team phát triển có đội ngũ kỹ thuật tốt, có nguồn tài nguyên phong phú, sẵn có để đảm bảo được yêu cầu, đọc nhanh, test nhanh và coding nhanh.
- Nếu khách hàng có sự tự tin cao trong yêu cầu thiết kế (nghĩa là ít thay đổi, ít dao động) thì V model là lựa chọn cần thiết.

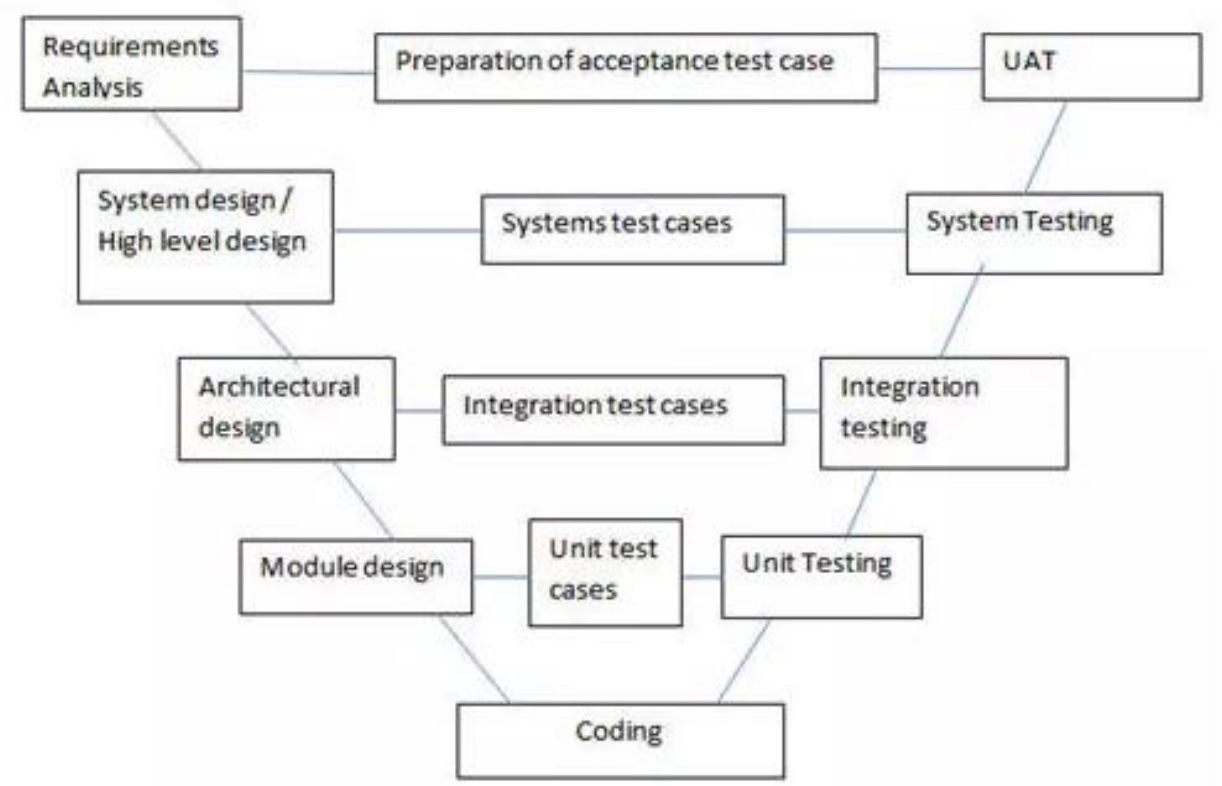
- Yêu cầu được xác định rõ ràng và không mơ hồ
- Tiêu chí chấp nhận được xác định rõ ràng.
- Dự án có quy mô vừa và nhỏ.
- Công nghệ và công cụ được sử dụng không thường xuyên thay đổi.

Mô hình V (Ưu điểm)



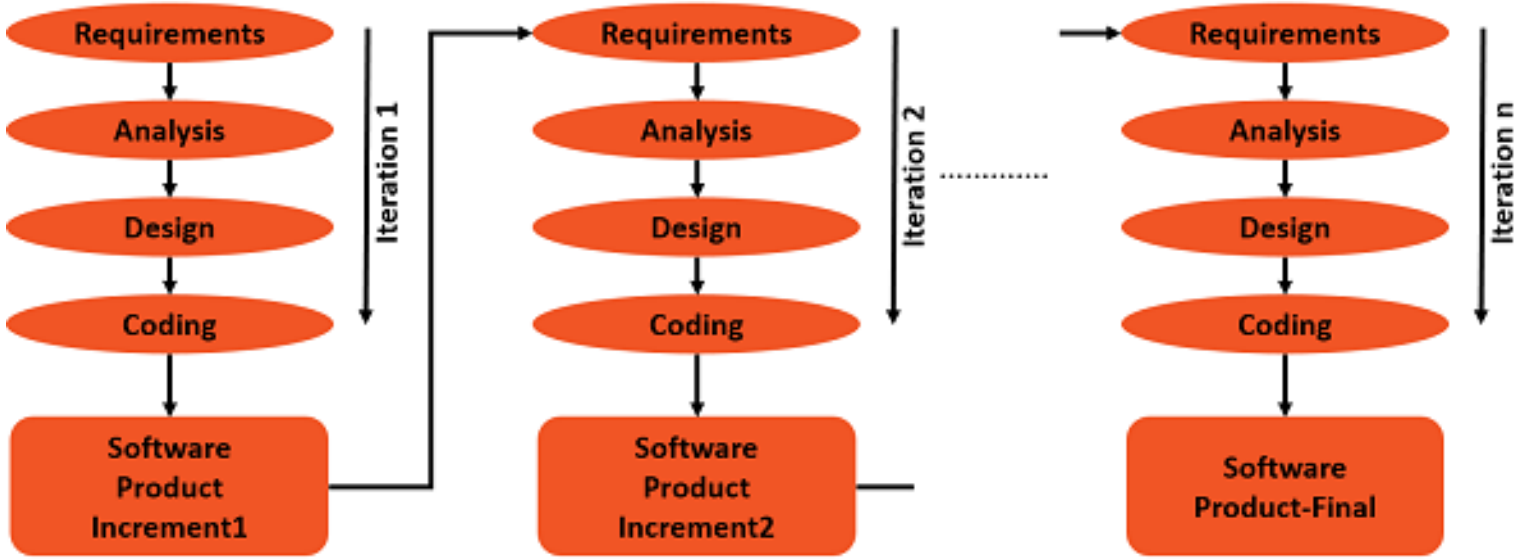
- Quá trình phát triển và quy trình quản lý có tính tổ chức và hệ thống
- Hoạt động tốt cho các dự án có quy mô vừa và nhỏ.
- Kiểm tra bắt đầu từ khi bắt đầu phát triển vì vậy sự mơ hồ được xác định ngay từ đầu.
- Dễ dàng quản lý vì mỗi giai đoạn có các mục tiêu và mục tiêu được xác định rõ ràng.

Mô hình V (Nhược điểm)



- Không thích hợp cho các dự án lớn và phức tạp
- Không phù hợp nếu các yêu cầu thường xuyên thay đổi.
- Không có phần mềm làm việc được sản xuất ở giai đoạn trung gian.
- Không có điều khoản cho việc phân tích rủi ro nên có sự không chắc chắn và có tính rủi ro.

Mô hình gia tăng lặp lại



- Trong mô hình gia tăng lặp lại, ban đầu, việc triển khai từng phần của một hệ thống tổng thể được xây dựng để nó ở trạng thái có thể phân phối.
- Tăng cường chức năng được thêm vào. Các khiếm khuyết, nếu có, từ lần giao hàng trước được sửa chữa và sản phẩm đang hoạt động được chuyển giao.
- Quá trình này được lặp lại cho đến khi hoàn thành toàn bộ quá trình phát triển sản phẩm. Sự lặp lại của các quá trình này được gọi là lặp lại.
- Vào cuối mỗi lần lặp lại, phần gia tăng sản phẩm sẽ được phân phối.

Mô hình gia tăng lặp lại

ƯU ĐIỂM

- Bạn có thể phát triển các yêu cầu ưu tiên trước.
- Việc giao sản phẩm ban đầu nhanh hơn.
- Khách hàng nhận được chức năng quan trọng sớm.
- Giảm chi phí giao hàng ban đầu.
- Mỗi lần phát hành là một bước tăng sản phẩm, do đó khách hàng sẽ luôn có trong tay một sản phẩm hoạt động.
- Khách hàng có thể cung cấp thông tin phản hồi cho từng sản phẩm, do đó tránh được những bất ngờ khi kết thúc quá trình phát triển.
- Các thay đổi yêu cầu có thể được điều chỉnh dễ dàng.

Mô hình gia tăng lặp lại

ƯU ĐIỂM

- Có thể sớm tạo ra nguyên mẫu của sản phẩm trong vòng đời phát triển của nó.
- Độ linh hoạt cao hơn và khi thay đổi yêu cầu dự án thì chi phí sẽ ít hơn nhiều, vì những thay đổi thuộc về module nào thì module đó sẽ thay đổi mà các module khác không hề bị ảnh hưởng.
- Việc phân chia thành các module cũng sẽ làm cho việc test nhẹ nhàng hơn, những module đơn giản thì test cũng đơn giản, sớm kết thúc.
- Giảm chi phí cho lần đầu giao sản phẩm.
- Dễ dàng quản lý các rủi ro có thể phát sinh.

Mô hình gia tăng lặp lại

KHUYẾT ĐIỂM

- Yêu cầu lập kế hoạch lặp lại hiệu quả.
- Cần phải có những khả năng thiết kế tốt và phương pháp tốt, để có thể hiểu rõ được yêu cầu và biết cách phân chi nó ra như thế nào cho hợp lý.
- Yêu cầu thiết kế hiệu quả để đảm bảo bao gồm các chức năng cần thiết và cung cấp cho các thay đổi sau này.
- Yêu cầu định nghĩa sớm về một hệ thống đầy đủ và đầy đủ chức năng để cho phép xác định các gia số.
- Tổng chi phí của hệ thống hoàn chỉnh không thấp hơn.

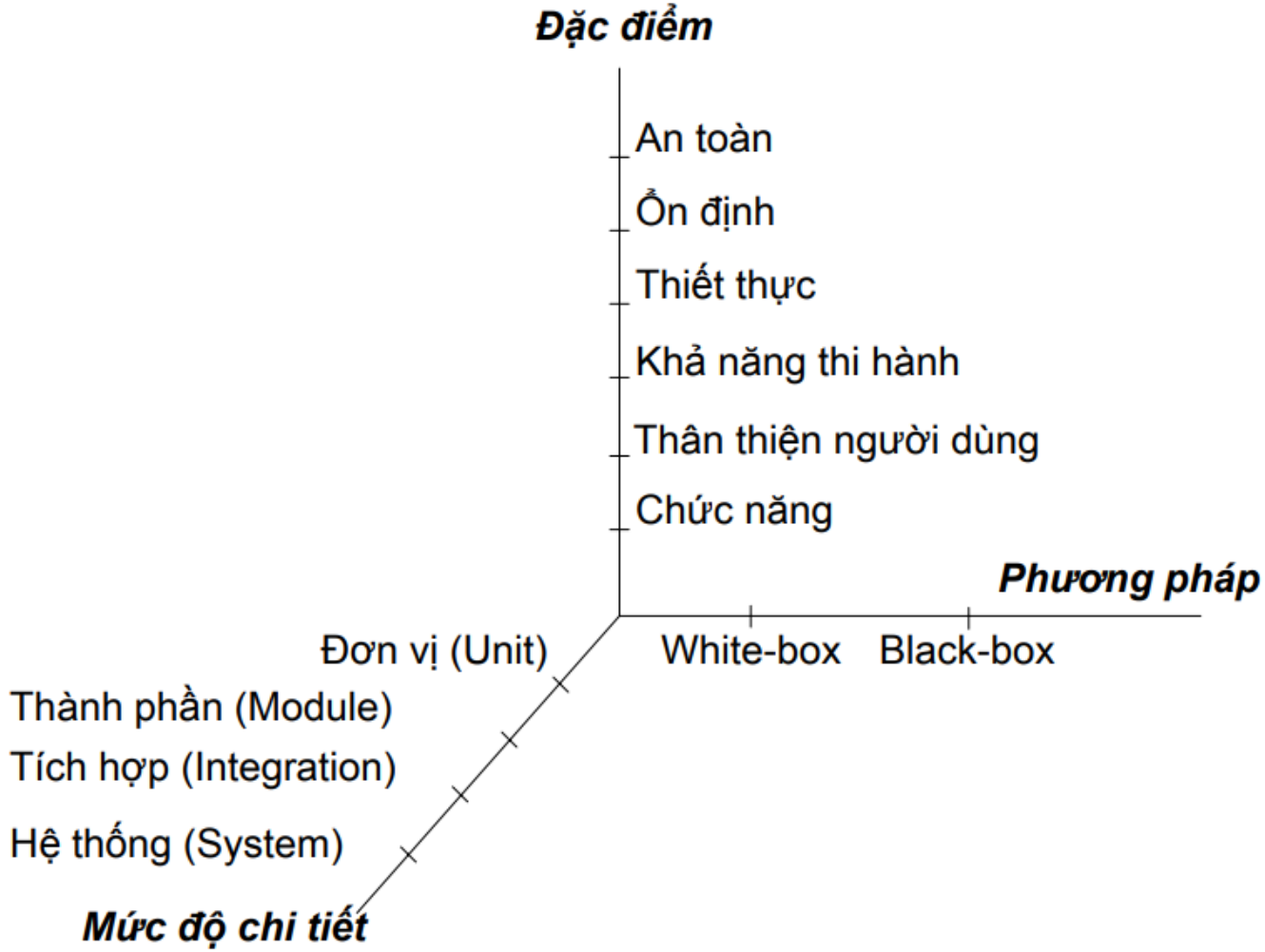
Kiểm tra trong mô hình phát triển phần mềm

- Đặc tính chung của một kiểm thử tốt
  - Kiểm thử cho mỗi giai đoạn/phần phát triển
  - Các mức kiểm tra nhấn mạnh vào mục tiêu phối hợp liên tục, không trùng lắp
  - Phân tích, thiết kế bắt đầu sớm, ngăn ngừa lỗi

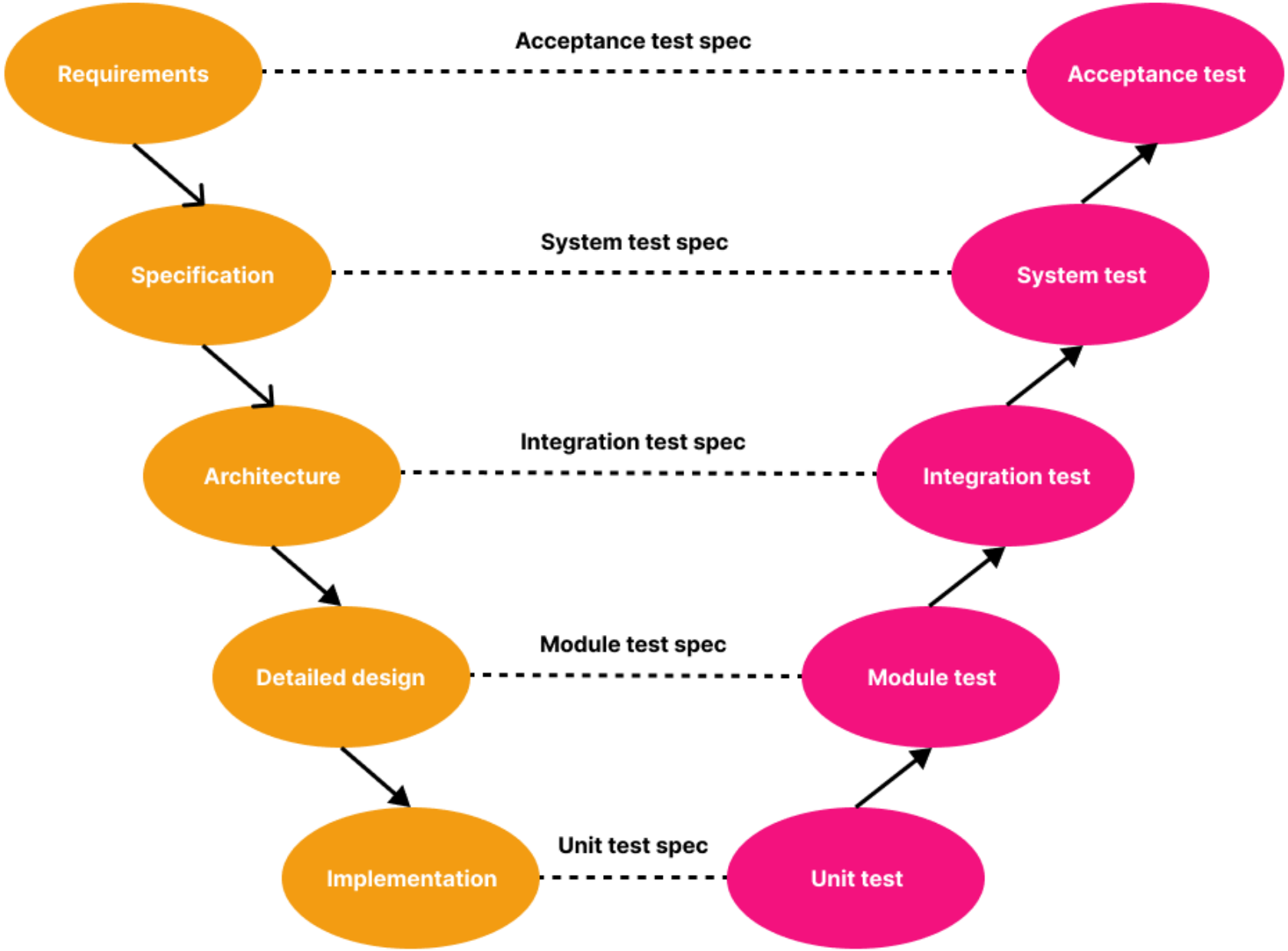
Kiểm chứng và chứng thực

- Kiểm chứng
  - Tìm các lỗi trong từng giai đoạn
  - các hành động để đảm bảo cho phần mềm được hiện thực đúng theo một chức năng cụ thể nào đó
- Chứng thực
  - Tìm lỗi trong hệ thống, phần chuyển giao
  - các hành động để đảm bảo cho phần mềm được xây dựng theo đúng yêu cầu của khách hàng

Các mức kiểm tra



Các mức kiểm tra



COMPONENT / UNIT TEST

Các mức kiểm tra

- Mục tiêu:
  - Tìm lỗi trong từng phần đơn lẻ
- Cơ sở:
  - mã, csdl, yêu cầu, thiết kế, các rủi ro chất lượng
- Kiểu kiểm tra:
  - Chức năng, sử dụng tài nguyên, hiệu suất, cấu trúc
- Công cụ:
  - Mức API (drivers , stubs), …
- Người thực hiện:
  - Người lập trình

COMPONENT / UNIT TEST

Các mức kiểm tra

- Tiêu biểu
  - Can thiệp mã
  - Thực thi trong môi trường độc lập
  - Yêu cầu drivers, stubs
  - Được thực hiện bởi người viết chương trình
- Thường được sửa trực tiếp, không lập báo cáo
- Phát triển theo hướng kiểm thử
  - Phát triển tập kiểm thử đơn vị
  - Xây dựng và tích hợp mã
  - Thực thi

INTEGRATION TEST

Các mức kiểm tra

- Mục tiêu:
  - Tìm lỗi trong mối quan hệ và giao diện giữa các cặp, nhóm thành phần liên quan
- Cơ sở:
  - Thiết kế, kiến trúc, giản đồ, lưu đồ. Rủi ro chất lượng
- Kiểu kiểm tra:
  - Chức năng, tài nguyên, hiệu năng
- Công cụ:
  - Mức API , CLI , ..
- Người thực hiện:
  - Cả người kiểm thử và người lập trình

INTEGRATION TEST

Các mức kiểm tra

- Nhiều hơn 1 mức kiểm tra tích hợp trong dự án
  - Tích hợp thành phần:
    - Tìm lỗi tương tác các thành phần
  - Tích hợp hệ thống:
    - Tìm lỗi tương tác trên toàn hệ thống
- Phức tạp
  - Nhiều tổ chức
  - Tiến trình nghiệp vụ
  - Độ tương thích Hardware/system

SYSTEM TEST

Các mức kiểm tra

- Mục tiêu:
  - Tìm lỗi trên toàn bộ và cá biệt về hành vi, chức năng, đáp ứng của hệ thống
- Cơ sở:
  - Yêu cầu, thiết kế mức cao, use cases, rủi ro, kinh nghiệm, môi trường, checklists
- Kiểu kiểm tra:
  - Chức năng, bảo mật, hiệu năng, tin cậy, khả dụng, khả chuyển, ...
- Công cụ:
  - API, CLI, GUI, …
- Người thực hiện:
  - Người kiểm tra độc lập

ACCEPTANCE TEST

Các mức kiểm tra

- Mục tiêu:
  - Chạy thử sản phẩm sẵn sàng cho triển khai/xuất xưởng
- Cơ sở:
  - Yêu cầu, hợp đồng, kinh nghiệm
- Kiểu kiểm tra:
  - Chức năng, khả chuyển, hiệu năng
- Công cụ:
  - GUI
- Người thực hiện:
  - Thường là Khách hàng/người sử dụng
  - Người kiểm tra độc lập

ACCEPTANCE TEST

Các mức kiểm tra

- Kiểm tra chấp nhận với NSD:
  - Người sử dụng nghiệp vụ xác nhận mức phù hợp cho mục cho chức năng,
- Kiểm tra tác vụ (*Operational testing*):
  - Chấp nhận bởi người quản trị
- Hợp đồng và kiểm tra quy tắc (*regulation testing*):
  - Kiểm chứng xác nhận hợp lệ theo hợp đồng.
- Kiểm tra Alpha, Beta, và lĩnh vực (*field testing*):
  - Kiểm thử và xây dựng tin cậy bởi các khách hàng đã có hay tiềm năng
  - Kiểm thử Beta và lĩnh vực được thực hiện trong môi trường thực

Các kiểu kiểm tra

KIỂM TRA CHỨC NĂNG

- Các hoạt động hợp lý, đã yêu cầu không được cung cấp, truy xuất, hư hỏng trầm trọng.
  - *Không thêm chức năng trong bảng tính*
  - *Hiện thực , không làm việc*
  - *Có thể chỉ số nguyên, không số thực*
- Hành động đúng, kết quả sai
  - Hàm ADD: 2+2=5?
- Hàm đúng, kết quả đúng .. Sai thể hiện
  - Hàm DIV: 2/2=I

Các kiểu kiểm tra

KIỂM TRA CHỨC NĂNG
PHI CHỨC NĂNG

- Bản địa hóa (*Localization : user interface, operational*)
- Chuẩn và đúng nguyên tắc (*regulatory compliance)*
- Xử lý lỗi và phục hồi
- Phục hồi tai họa (*Disaster recovery*)
- Thực hiện và phân bố trên mạng
- Thời gian , sắp xếp
- Chất lượng dữ liệu
- Hoán chuyển dữ liệu
- Tác vụ
- Cài đặt
- Phân giải cài đặt
- Xử lý ngày tháng
- Tài liệu
- …

Các kiểu kiểm tra

KIỂM TRA CẤU TRÚC

- Trên cơ sở hệ thống được xây dựng như thế nào ?
  - Mã
  - Dữ liệu
  - Thiết kế
- Bao phủ Cấu trúc (*white box*) có thể được đánh giá chức năng và phi chức năng non-functional (*black box*)

Các kiểu kiểm tra

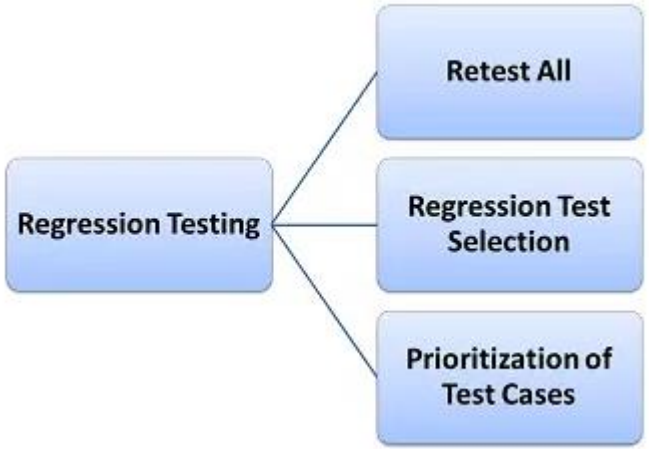
KIỂM TRA HỒI QUY VÀ XÁC NHẬN

- Chẳng hạn, một ứng dụng phần mềm có thể cho phép giáo viên thêm, lưu, xóa và làm mới trong một công cụ học tập trực tuyến. Sau đó, các Developer deploy 1 version mới với chức năng bổ sung để cập nhật. Chức năng mới được kiểm tra để xác nhận rằng bản cập nhật hoạt động như mong đợi. Trong trường hợp này, kiểm tra hồi quy có thể cải thiện chất lượng tổng thể của sản phẩm.
- Sau khi phần mềm thay đổi, điều quan trọng là đảm bảo bạn không làm hỏng bất cứ thứ gì. Ngay cả sau khi bạn phát hiện và sửa lỗi hồi quy, vẫn cần phải thử nghiệm thêm. Bạn cần xác nhận rằng việc xử lý một lỗi không gây ra các vấn đề khác.
- Thử nghiệm hồi quy là cần thiết, ngay cả khi thực hiện các thay đổi rất nhỏ đối với code.
- Kiểm tra hồi quy cũng thường được thực hiện bất cứ khi nào một vấn đề được phát hiện trước đó đã được khắc phục. Sau tất cả, có nhiều người phụ thuộc kiểm tra khi nào hoặc không mã mới tuân thủ mã cũ và mã không được sửa đổi đó không bị ảnh hưởng.

Các kiểu kiểm tra

CÁC PHƯƠNG PHÁP KIỂM TRA HỒI QUY

- Kiểm tra lại tất cả: Điều này giúp kiểm tra lại tính toàn vẹn của phần mềm từ trên xuống dưới.
- Lựa chọn kiểm tra hồi quy: Phương pháp này cho phép nhóm chọn một lựa chọn đại diện cho các thử nghiệm sẽ xấp xỉ một thử nghiệm đầy đủ. Điều này đòi hỏi ít thời gian hoặc chi phí hơn nhiều so với bộ thử nghiệm đầy đủ
- Ưu tiên cho trường hợp thử nghiệm: Với các trường hợp thử nghiệm hạn chế, hãy thử ưu tiên các thử nghiệm có thể ảnh hưởng đến cả bản dựng hiện tại và tương lai của phần mềm.



Các kiểu kiểm tra

CÁC PHƯƠNG PHÁP KIỂM TRA HỒI QUY

- Kiểm tra hồi quy (*Regression testing*)
  - Kiểm tra ảnh hưởng của thay đổi dù cho nó có nhỏ, cục bộ, biệt lập
- Kiểm tra xác nhận (*Confirmation testing*) xác nhận
  - Thay đổi hệ thống đã có
  - Lỗi đã cố định phải hỗ trợ theo dõi
- Luôn lặp lại kiểm định
- Tự động, kiểm tra sâu, có ích