HUTECH
Đại học Công nghệ Tp.HCM

KIỂM THỬ PHẦN MỀM

Bài 1:
TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM

Thời gian: 5 tiết

NỘI DUNG

1.1 Các mô hình phát triển phần mềm

1.2 Kiểm thử phần mềm

1.3 Vai trò của kiểm thử phần mềm

1.1 CÁC MÔ HÌNH PHÁT TRIỂN PHẦN MỀM

1.1.1 Waterfall Model

1.1.2 Prototype Model

1.1.3 Spiral Model

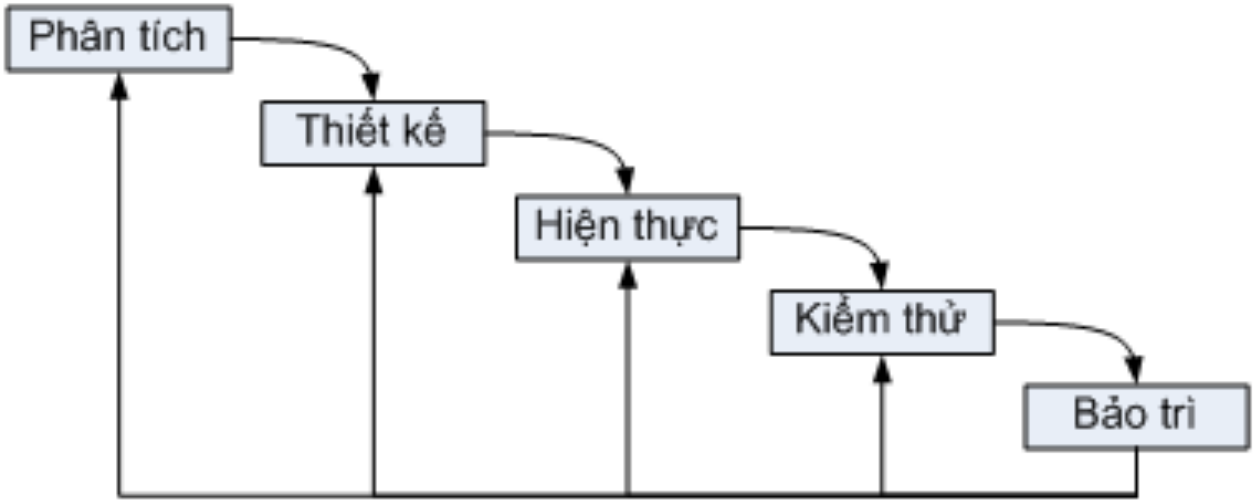
1.1.4 SDLC Model

1.1.5 Agile Model

1.1.7 V- Model

1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

- ✓ *Mô hình tuần tự tuyến tính.* Một trong các mô hình đầu tiên và phổ biến.
- ✓ Chia quá trình phát triển PM thành những giai đoạn tuần tự nối tiếp. Mỗi giai đoạn có mục đích nhất định. Kết quả của giai đoạn trước là đầu vào cho giai đoạn tiếp theo sau. Mô hình thác nước có 5 giai đoạn:



Hình 1.1: Mô hình thác nước

1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

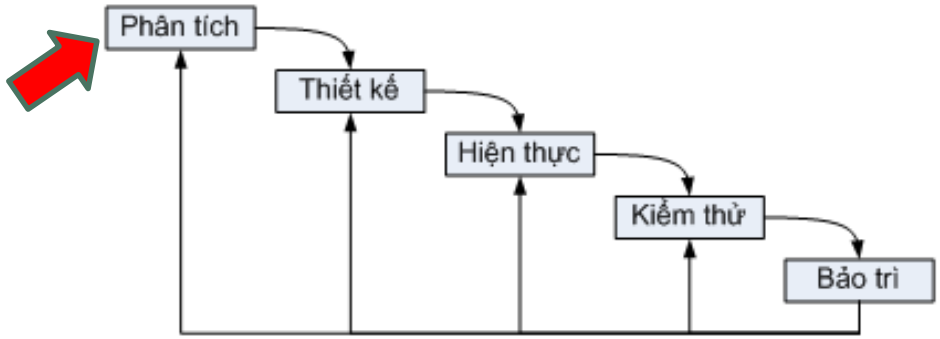
***Xác định yêu cầu*:** Tiến hành khi có nhu cầu xây dựng PM.

- ✓ Mục tiêu: Xác định chính xác các yêu cầu cho P.Mềm.
- ✓ Kết quả nhận: Thông tin về hoạt động của thế giới thực.
- ✓ Kết quả chuyển giao: Các yêu cầu (công việc sẽ thực hiện trên máy tính) cùng với các thông tin mô tả chi tiết về các yêu cầu (cách thức thực hiện)

1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

***Phân tích*: Tiến hành ngay sau việc xác định yêu cầu.**

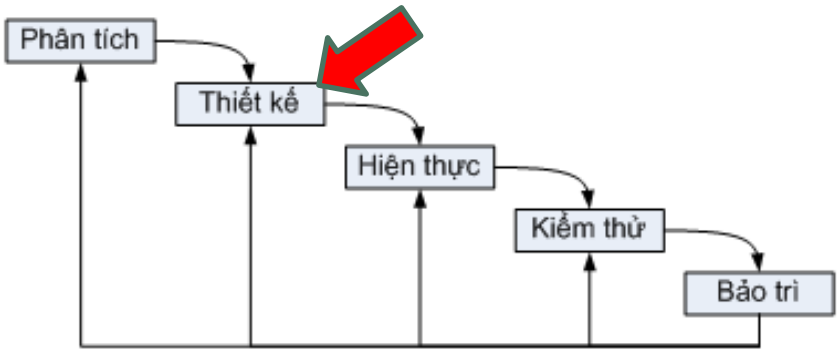
- ✓ Mục tiêu: Mô tả yêu cầu bằng mô hình
- ✓ Kết quả nhận: Các yêu cầu cùng thông tin liên quan.
- ✓ Kết quả chuyển giao:
  - Mô hình xử lý (các công việc và Quan hệ)
  - Mô hình dữ liệu (các thông tin được sử dụng và QH)
  - Mô hình khác (không gian, thời gian, con người…)



1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

Thiết kế: Tiến hành sau khi kết thúc việc phân tích.

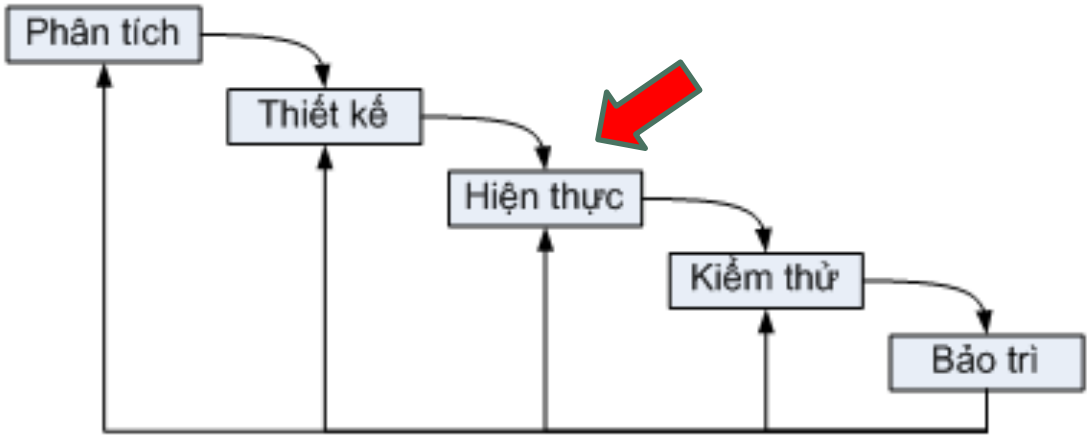
- ✓ Mục tiêu: Mô tả các thành phần của phần mềm.
- ✓ Kết quả nhận: Mô hình thế giới thực.
- ✓ Kết quả chuyển giao:
  - Mô tả thành phần giao diện: Các hàm/CTDL.
  - Mô tả thành phần xử lý: Các hàm kiểm tra xử lý.
  - Mô tả thành phần dữ liệu: Các hàm đọc/ghi...



1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

***Hiện thực*:** Tiến hành ngay sau kết thúc việc thiết kế.

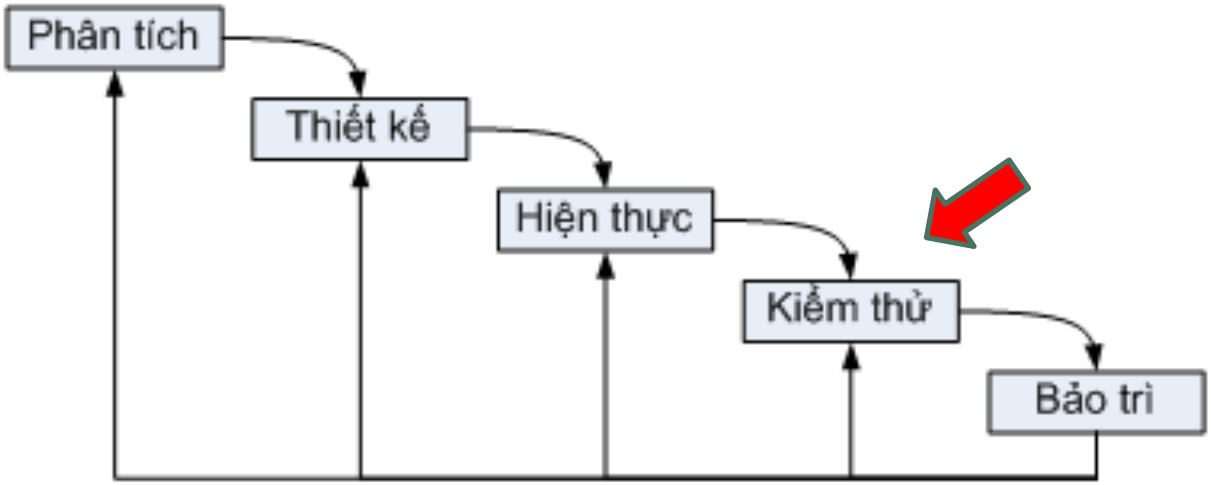
- ✓ Mục tiêu: Tạo lập phần mềm theo yêu cầu.
- ✓ Kết quả nhận: Mô hình phần mềm.
- ✓ Kết quả chuyển giao: Chương trình nguồn của phần mềm với CTDL tương ứng và chương trình thực hiện.



1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

***Kiểm thử*: Tiến hành sau kết quả hiện thực (lập trình).**

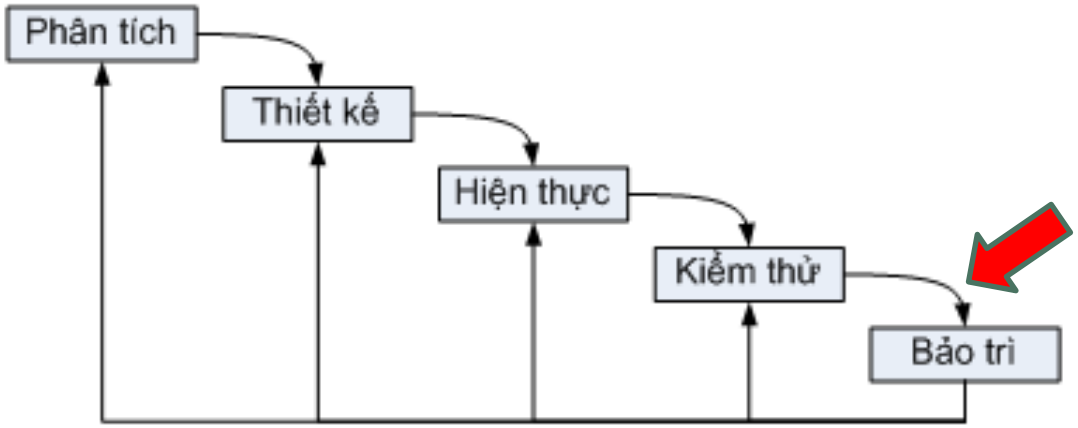
- ✓ Mục tiêu: Tăng độ tin cậy của phần mềm.
- ✓ Kết quả nhận: các yêu cầu, mô hình phần mềm,…
- ✓ Kết quả chuyển giao: Phần mềm có độ tin cậy cao.



1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

***Bảo trì*:** Công việc của giai đoạn bao gồm việc cài đặt và vận hành phần mềm trong thực tế.

- ✓ Mục tiêu: Đảm bảo phần mềm vận hành tốt
- ✓ Kết quả nhận: Phần mềm đã hoàn thành
- ✓ Kết quả chuyển giao: Các phản ánh của khách hàng trong quá trình sử dụng.



1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

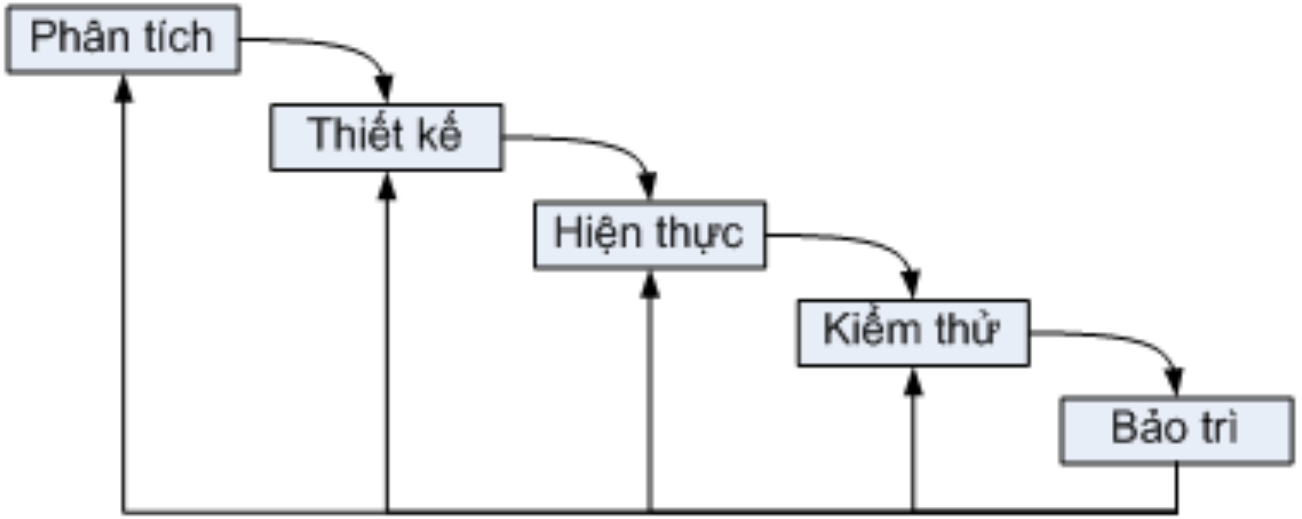
Nhận xét:

- ✓ Dễ phân chia quá trình xây dựng PM thành những giai đoạn độc lập.
- ✓ Các dự án lớn ít khi tuân theo dòng chảy tuần tự của mô hình (cần lặp lại các bước để nâng chất lượng, khách hàng ít tuyên bố hết các yêu cầu trong giai đoạn phân tích).
- ✓ Rất khó thay đổi khi đã thực hiện xong một giai đoạn, khó thay đổi các yêu cầu theo ý khách hàng.
- ✓ Phương pháp này chỉ thích hợp đã hiểu rất rõ các yêu cầu của khách hàng, những thay đổi sẽ được giới hạn.

1.1.1. MÔ HÌNH THÁC NƯỚC Waterfall Model

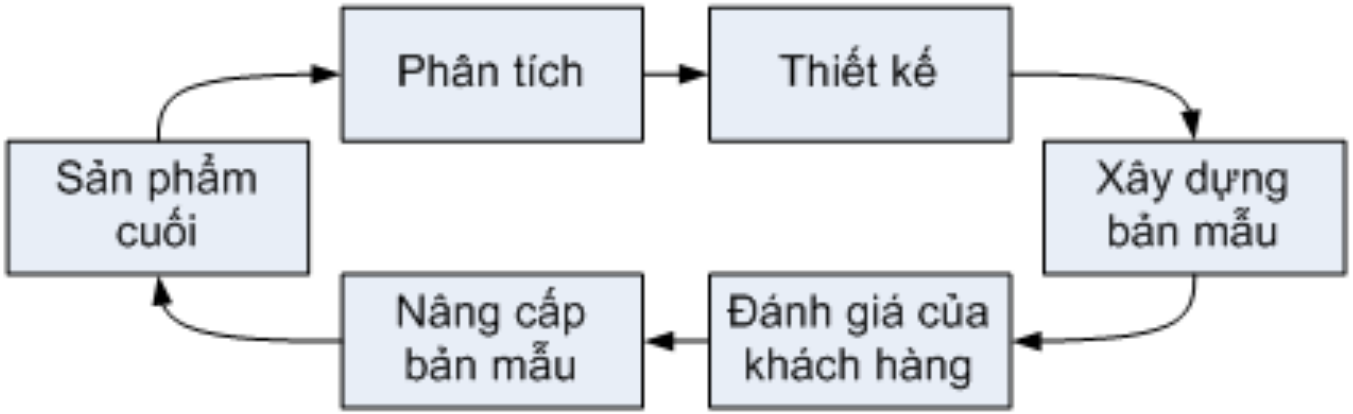
Chú ý:

Mô hình thác nước có thể được cải tiến bằng cách cho phép quay lui khi phát hiện lỗi trong giai đoạn phía trước.



1.1.2. MÔ HÌNH BẢN MẪU Prototype Model

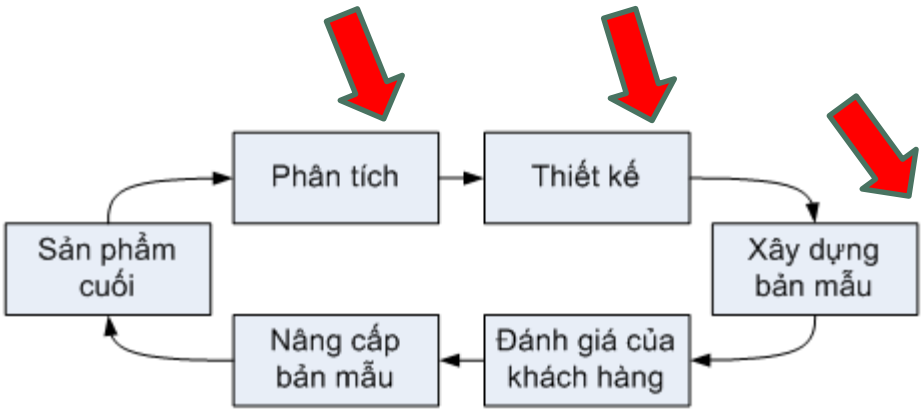
- ✓ Tương tự như mô hình thác nước với việc bổ sung vào *giai đoạn thực hiện phần mềm mẫu*.
- ✓ Có thể tiến hành lặp lại mà không nhất thiết theo trình tự nhất định.



Hình 1.2: Mô hình bản mẫu

1.1.2. MÔ HÌNH BẢN MẪU Prototype Model

- ✓ Ngay sau giai đoạn Xác định yêu cầu phân tích, sẽ đưa ra một bản thiết kế sơ bộ
- ✓ Tiếp theo tiến hành hiện thực bản mẫu đầu tiên, rồi chuyển cho người sử dụng.
- ✓ Bản mẫu này chỉ nhằm để mô tả cách thức phần mềm hoạt động cũng như cách người dùng tương tác.

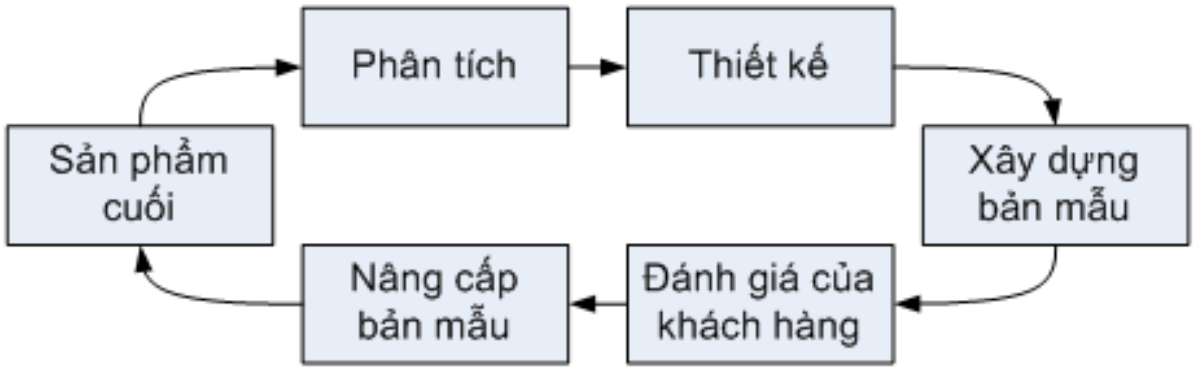


1.1.2. MÔ HÌNH BẢN MẪU Prototype Model

- ✓ Người dùng sau khi xem xét sẽ phản hồi thông tin cần thiết lại cho nhóm phát triển.
- ✓ Nếu người dùng đồng ý với bản mẫu, nhóm phát triển sẽ tiến hành hiện thực. Ngược lại, phải quay lại giai đoạn xác định yêu cầu.
- ✓ Lặp lại liên tục cho đến khi người sử dụng đồng ý với bản mẫu do nhà phát triển đưa ra.

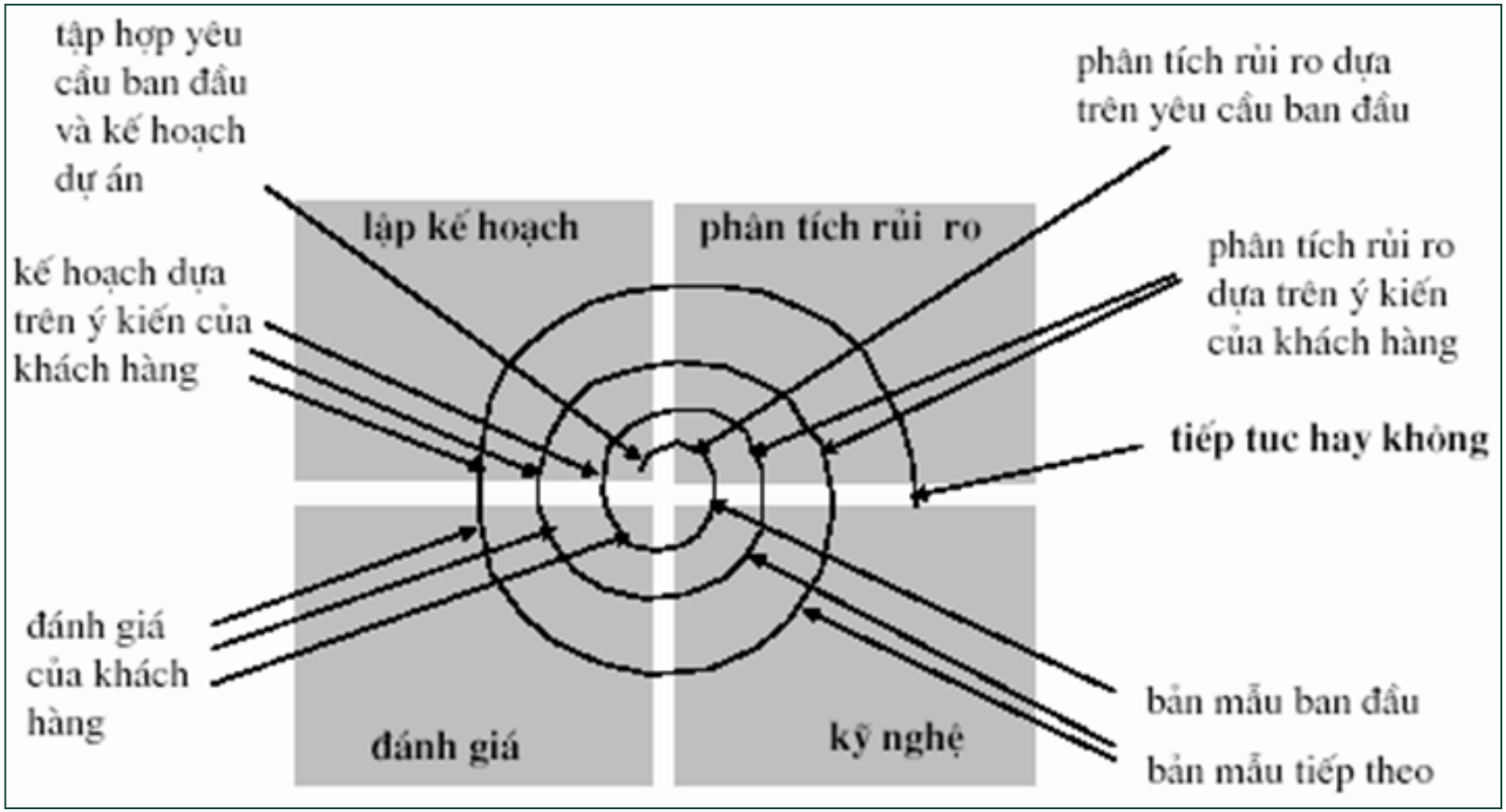
1.1.2. MÔ HÌNH BẢN MẪU Prototype Model

- ✓ Là một hướng tiếp cận tốt khi các yêu cầu chưa rõ ràng.
- ✓ Tính cấu trúc không cao dễ mất tin tưởng của khách hàng, và thiếu tầm nhìn của cả quy trình;
- ✓ Chỉ nên áp dụng với những hệ thống có tương tác ở mức độ nhỏ/vừa; một phần của hệ thống lớn; hoặc có thời gian chu kỳ tồn tại ngắn.



1.1.3. MÔ HÌNH XOẮN ỐC Spiral Model

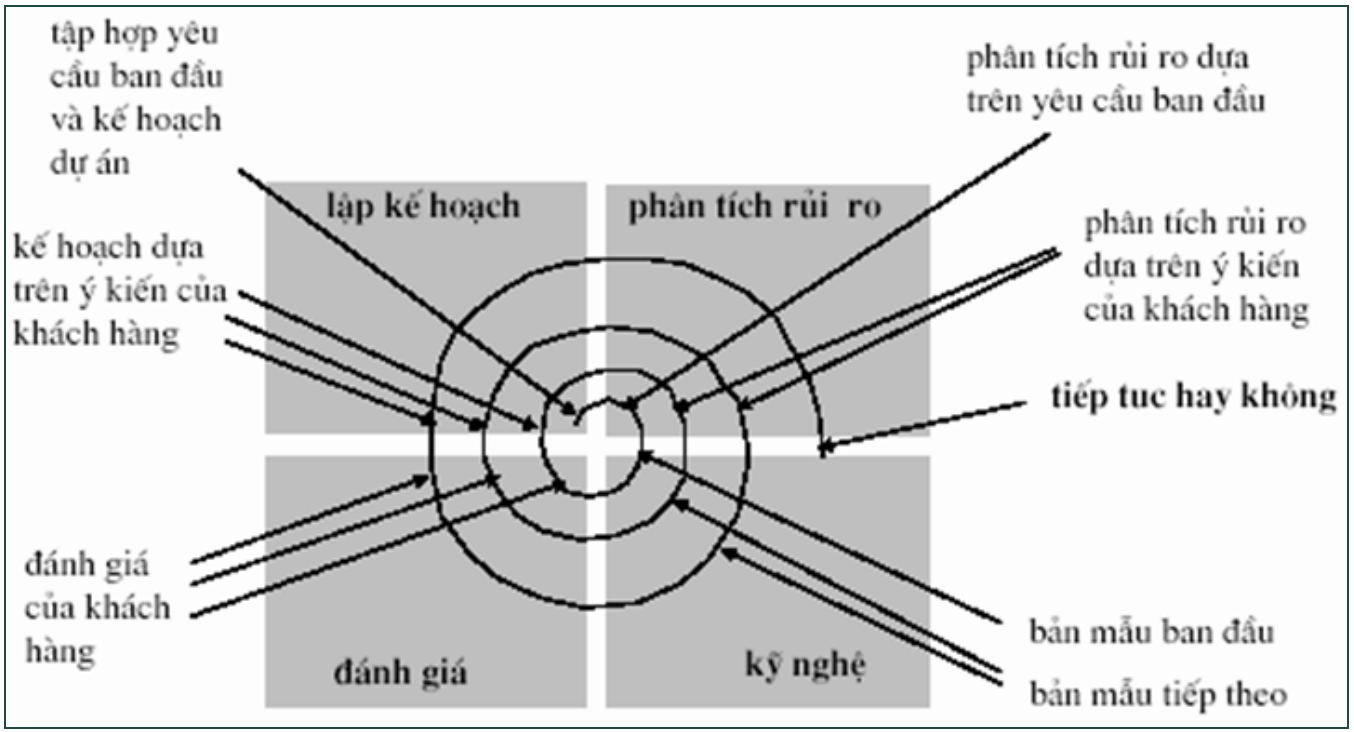
- ✓ Là sự *kết hợp của mô hình bản mẫu thiết kế và mô hình thác nước* được *lặp lại nhiều lần*.



Hình 1.3: Mô hình xoắn ốc

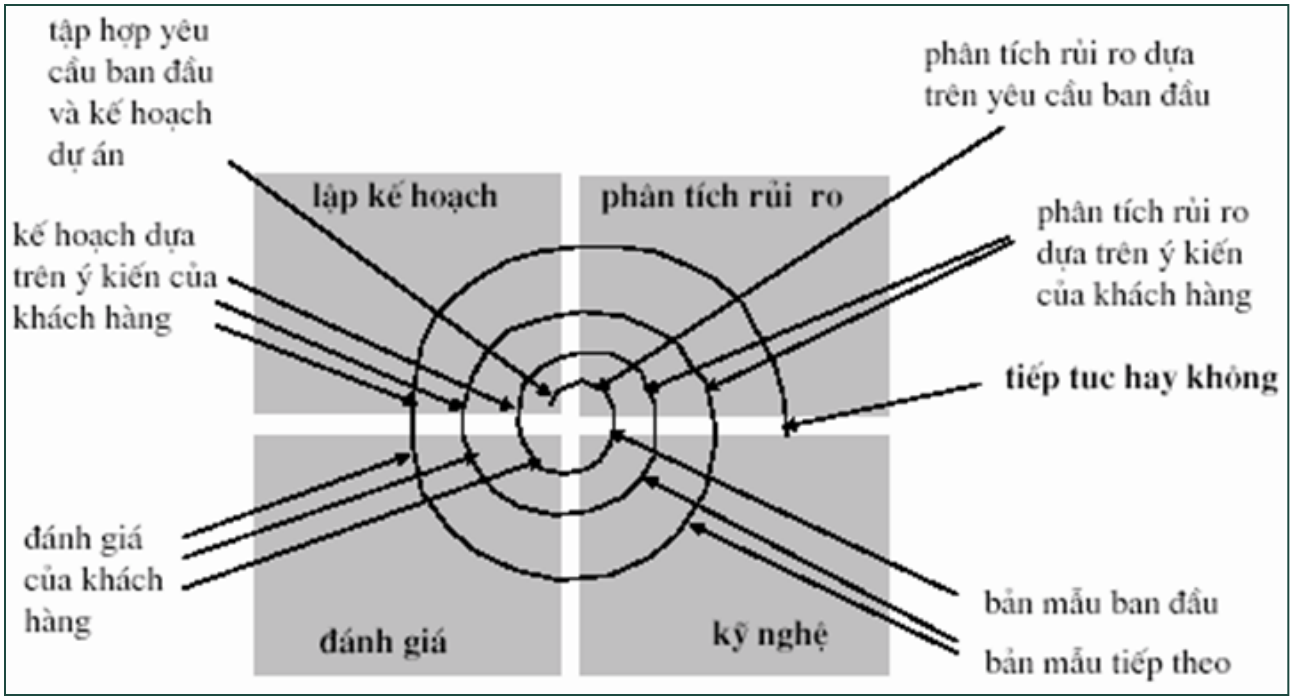
1.1.3. MÔ HÌNH XOẮN ỐC Spiral Model

- ✓ Ở lần lặp tiếp theo, hệ thống sẽ được tìm hiểu và xây dựng hoàn thiện hơn ở lần lặp trước đó
- ✓ Yêu cầu của người dùng ngày càng rõ ràng hơn, và các bản mẫu phần mềm ngày một hoàn thiện hơn.



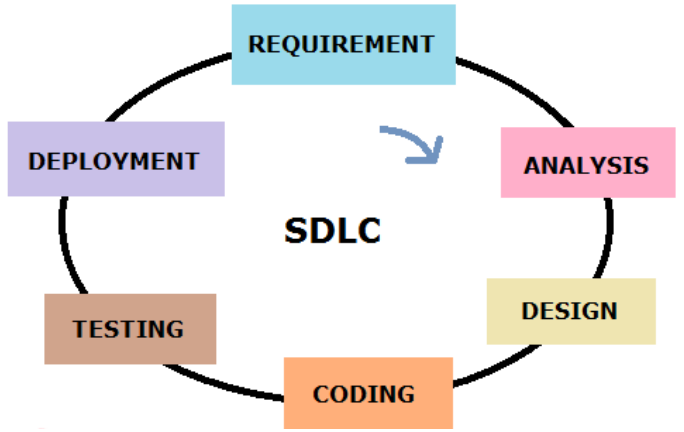
1.1.3. MÔ HÌNH XOẮN ỐC Spiral Model

- ✓ Ngoài ra, ở cuối mỗi lần lặp sẽ có thêm công đoạn phân tích mức độ rủi ro để quyết định xem có nên đi tiếp theo hướng này hay không.
- ✓ Mô hình này phù hợp với các hệ thống phần mềm lớn do có khả năng kiểm soát rủi ro ở từng bước tiến hóa.



1.1.4. VÒNG ĐỜI PHÁT TRIỂN PHẦN MỀM- SDLC

- ✓ Software Development Life Cycle - SDLC là tập hợp các hoạt động nhằm tạo ra một hệ thống chất lượng, đáp ứng mong đợi của khách hàng.
- ✓ Một quy trình tốt và hợp lý luôn tạo ra những sản phẩm đạt tiêu chuẩn.
- ✓ Quy trình phát triển phần mềm đem lại chất lượng, năng suất, giá thành, …



1.1.5. MÔ HÌNH PHÁT TRIỂN LINH HOẠT AGILE MODEL

1.1.6. MÔ HÌNH HỢP NHẤT RUP MODEL

1.1.7. MÔ HÌNH CHỮ V – V MODEL

1.2. KIỂM THỬ PHẦN MỀM

1.2.1 Phần mềm và chất lượng phần mềm

1.2.2 Các yếu tố ảnh hưởng đến chất lượng phần mềm

1.2.3 Khái niệm kiểm thử phần mềm

1.2.4 Mục tiêu kiểm thử

1.2.5 Tầm quan trọng của kiểm thử

1.2.6 Các nguyên tắc trong kiểm thử

1.2.7 Một số khái niệm liên quan kiểm thử

1.2.8 Các đối tượng thực hiện kiểm thử

1.2.9 Các điểm cần lưu ý khi kiểm thử

1.2.10 Các hạn chế của kiểm thử

1.2.1 PHẦN MỀM VÀ CHẤT LƯỢNG PHẦN MỀM

Phần mềm

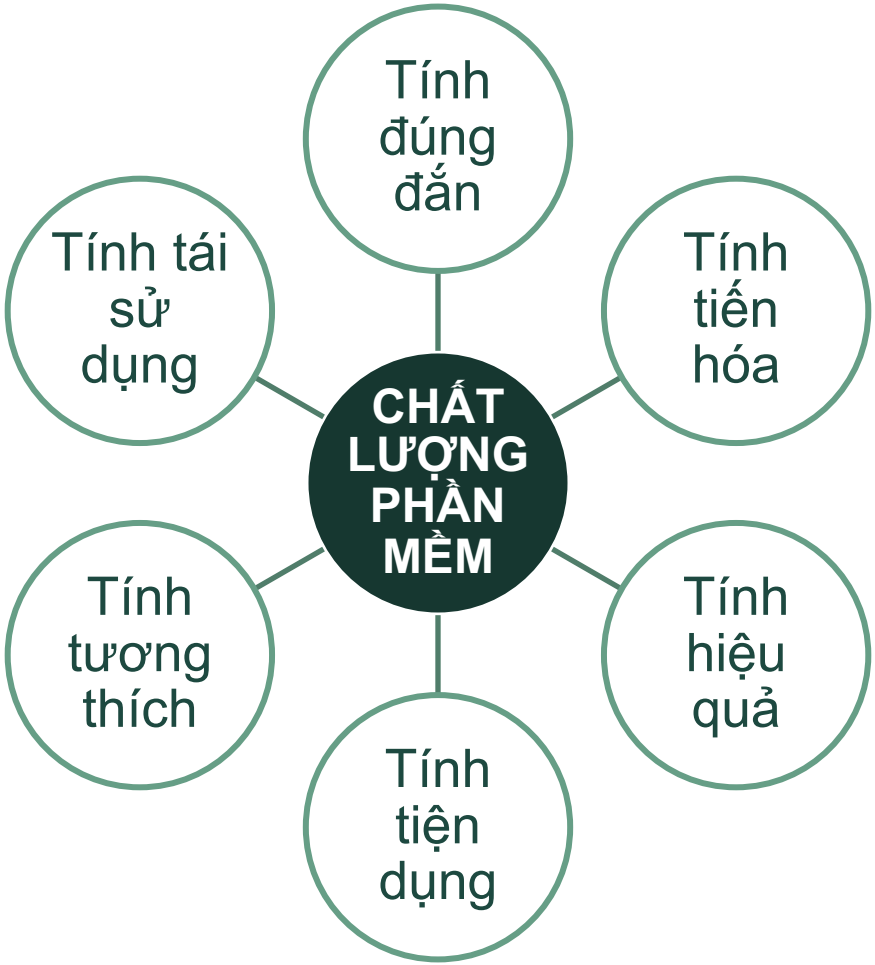
- ✓ *Chương trình máy tính*: Các chỉ thị để máy tính làm việc.
- ✓ *Phần mềm*: Các chương trình hỗ trợ thực hiện công iệc theo lĩnh vực chuyên ngành.
- ✓ *Nhiệm vụ chính yếu của phần mềm*: Thực hiện các công việc dễ dàng và nhanh chóng.
- ✓ *Hoạt động của phần mềm*: Mô phỏng lại các hoạt động của thế giới thực.
- ✓ *Quá trình sử dụng một phần mềm*: Thực hiện các công việc trên máy tính để hoàn tất công việc.

1.2.1 PHẦN MỀM VÀ CHẤT LƯỢNG PHẦN MỀM

Chất lượng phần mềm

«Phần mềm tốt là phần mềm phải đáp ứng các chức năng theo yêu cầu, có hiệu năng tốt, có khả năng bảo trì, đáng tin cậy, và được người sử dụng chấp nhận».

1.2.2 CÁC YẾU TỐ ẢNH HƯỞNG ĐẾN CHẤT LƯỢNG PM



1.2.2 CÁC YẾU TỐ ẢNH HƯỞNG ĐẾN CHẤT LƯỢNG PM

Tính đúng đắn:

✓ Thực hiện đầy đủ và chính xác các yêu cầu.

Tính đúng đắn xác định trên cơ sở:

- Tính đúng đắn của giải pháp xử lý / thuật toán,
- Tính đúng đắn của tập mã lệnh hoặc nội dung của chương trình,
- Tính đúng đắn qua kiểm thử, việc áp dụng chương trình trong một khoảng thời gian dài, trên diện rộng và với tần suất sử dụng cao,
- Tính tương đương của chương trình với thuật toán,

1.2.2 CÁC YẾU TỐ ẢNH HƯỞNG ĐẾN CHẤT LƯỢNG PM

Tính tiến hóa

- ✓ Sản phẩm có thể mở rộng, tăng cường về mặt chức năng dễ dàng,
- ✓ Người dùng khai báo thay đổi về quy định với phần mềm tùy theo thay đổi trong thế giới thực liên quan (như thay công thức tiền phạt …)

1.2.2 CÁC YẾU TỐ ẢNH HƯỞNG ĐẾN CHẤT LƯỢNG PM

Tính hiệu quả:

- ✓ Hiệu quả *kinh tế, ý nghĩa, giá trị* thu được.
- ✓ Hiệu quả *sử dụng* (tốc độ xử lý của phần mềm …)
- ✓ Hiệu quả *kỹ thuật* (tối ưu tài nguyên của máy tính: CPU, bộ nhớ, không gian xử lý …)

1.2.2 CÁC YẾU TỐ ẢNH HƯỞNG ĐẾN CHẤT LƯỢNG PM

Tính tiện dụng:

- ✓ Tính cơ động và linh hoạt của sản phẩm
- ✓ Cảm nhận (về mặt tâm lý) của người dùng về:
  - Dễ học, có giao diện trực quan tự nhiên.
  - Các *chức năng* của sản phẩm *dễ thao tác* …

1.2.2 CÁC YẾU TỐ ẢNH HƯỞNG ĐẾN CHẤT LƯỢNG PM

Tính tương thích:

- ✓ *Khả năng trao đổi dữ liệu với các phần mềm khác* (như: nhận danh sách nhân viên từ tập tin Excel …),
- ✓ Gồm Giao tiếp nội bộ và giao tiếp bên ngoài.

1.2.2 CÁC YẾU TỐ ẢNH HƯỞNG ĐẾN CHẤT LƯỢNG PM

Tính tái sử dụng:

- ✓ *Có thể áp dụng* cho *nhiều lĩnh vực* theo *nhiều chế độ làm việc* khác nhau,
- ✓ Áp dụng về mặt kỹ thuật hay phối hợp về mặt sử dụng với các phần mềm khác.

1.2.3 KHÁI NIỆM KIỂM THỬ PHẦN MỀM

- ✓ KTPM Là thí nghiệm để so sánh kết quả thực tế với lý thuyết nhằm phát hiện lỗi.
- ✓ Bộ thử nghiệm (test cases) là dữ liệu dùng để kiểm tra.
- ✓ Bộ kiểm thử tốt là bộ có khả năng phát hiện ra lỗi.
- ✓ KTPM chỉ chứng minh được sự tồn tại của lỗi nhưng không chứng minh được chương trình không có lỗi.

1.2.3 KHÁI NIỆM KIỂM THỬ PHẦN MỀM

Thực tế:

- ✓ Không gian kiểm thử: Các bộ kiểm thử (rất lớn).
- ✓ Nếu vét cạn được thì chương trình không còn lỗi.

  ➔ Không khả thi.
- ✓ Do đó tính đúng đắn của PM dùng độ tin cậy.
- ✓ PP kiểm thử là chọn bộ kiểm thử để độ tin cậy:
  - Phân hoạch không gian kiểm thử thành nhiều miền rồi chọn số liệu kiểm thử.
  - Cần tránh mọi bộ thử nghiệm đều rơi vào một miền kiểm tra.

1.2.4 MỤC TIÊU KIỂM THỬ

Các mục tiêu trực tiếp:

- ✓ Xác định và phát hiện nhiều lỗi nhất có thể
- ✓ Sau khi sửa chữa các lỗi và kiểm tra lại.
- ✓ Thực hiện các yêu cầu kiểm thử hiệu quả, trong phạm vi ngân sách và thời gian cho phép.

1.2.4 MỤC TIÊU KIỂM THỬ

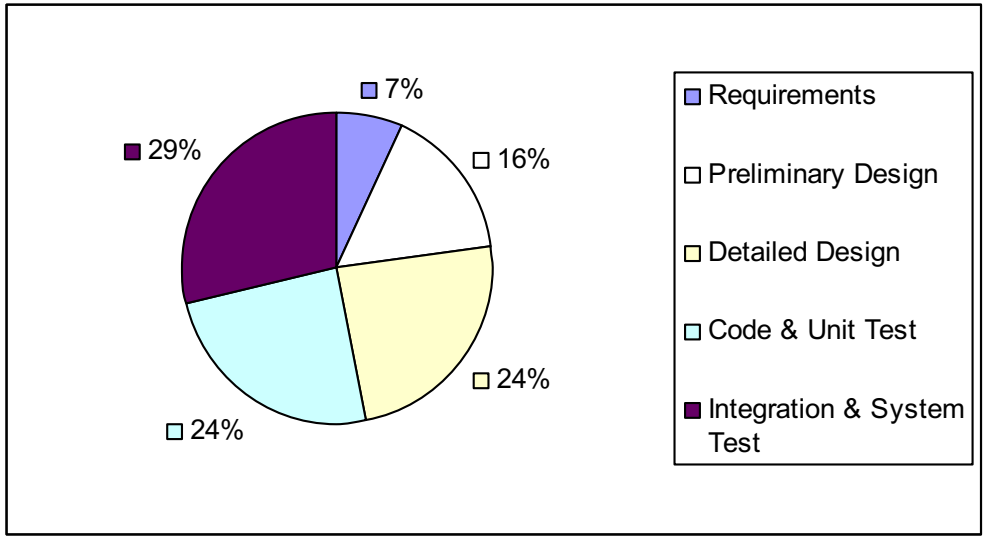
Các mục tiêu gián tiếp:

Tuỳ thuộc vào mục tiêu của kiểm thử, chia các mức

- ✓ Mức 0: testing và debuging là giống nhau
- ✓ Mức 1: Chỉ ra phần mềm hoạt động
- ✓ Mức 2: Chỉ ra phần mềm không hoạt động
- ✓ Mức 3: Giảm các rủi ro khi sử dụng phần mềm
- ✓ Mức 4: Phát triển phần mềm có chất lượng cao hơn.

1.2.5 TẦM QUAN TRỌNG CỦA KIỂM THỬ

- ✓ Chi phí cho quá trình kiểm thử và tích hợp hệ thống chiếm khoảng 29% tổng kinh phí của dự án
- ✓ Nhiều nhất trong số những quá trình còn lại bao gồm thiết kế ban đầu, thiết kế chi tiết, viết mã chương trình và kiểm thử từng chức năng, xác định yêu cầu.



Chi phí phân bố cho các tác vụ trong phát triển phần mềm

1.2.5 TẦM QUAN TRỌNG CỦA KIỂM THỬ

- ✓ Nghiên cứu bởi NIST trong năm 2002, các lỗi phần mềm gây tổn thất kinh tế Mỹ 59,5 tỷ USD.
- ✓ Hơn 1/3 chi phí này có thể tránh, nếu việc kiểm thử được thực hiện tốt hơn.
- ✓ Một khiếm khuyết nếu tìm ra sớm thì chi phí để sửa chữa sẽ tiết kiệm hơn nhiều.

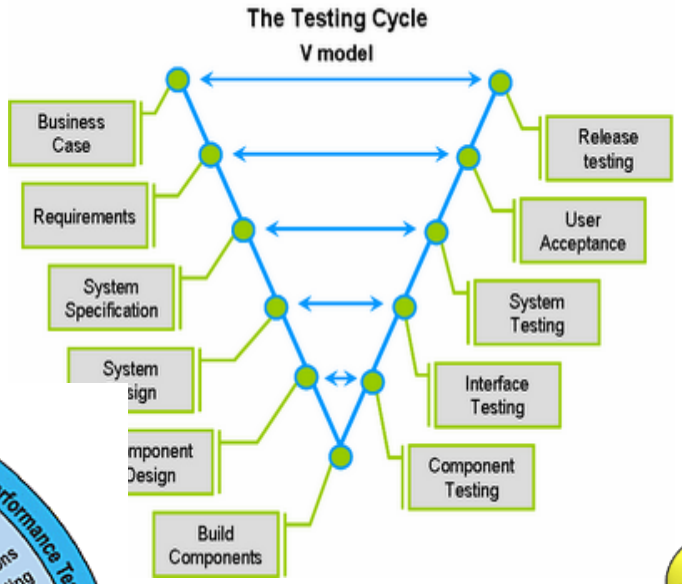
1.2.5 TẦM QUAN TRỌNG CỦA KIỂM THỬ

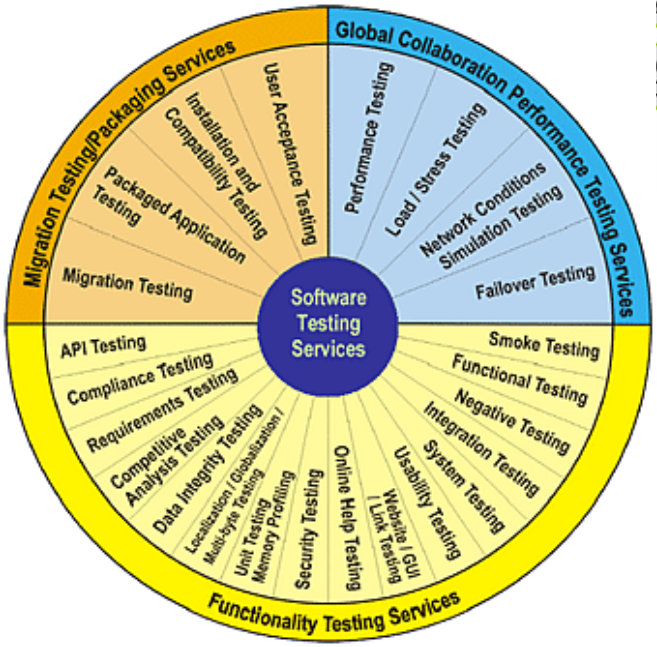
- ✓ Vấn đề tìm thấy sau khi ra bản chính thức có chi phí gấp 10-100 lần khi giải quyết lúc tiếp nhận yêu cầu.

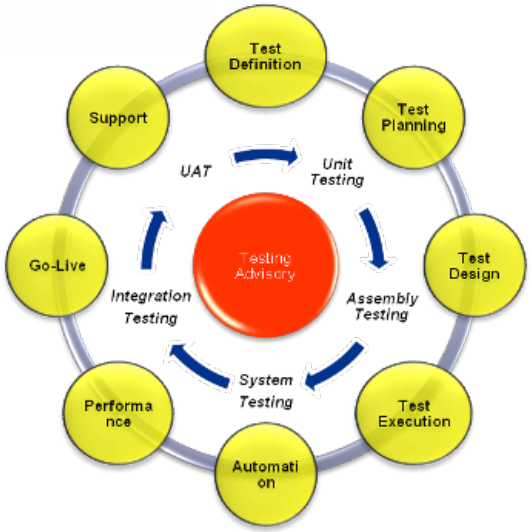
Chi phí sửa chữa một khiếm khuyết		*Thời gian phát hiện*				
		Yêu cầu phần mềm	*Kiến trúc phần mềm*	*Xây dựng phần mềm*	*Kiểm thử hệ thống*	*Sau khi phát hành*
Thời gian sử dụng	Yêu cầu phần mềm	1×	3×	5–10×	10×	10–100×
	Kiến trúc phần mềm	–	1×	10×	15×	25–100×
	Xây dựng phầm mềm	–	–	1×	10×	10–25×

Bảng chi phí sửa chữa khiếm khuyết tùy vào giai đoạn được tìm ra.

1.2.5 TẦM QUAN TRỌNG CỦA KIỂM THỬ







Các tác vụ trong quản lý chất lượng dự án

1.2.6 CÁC NGUYÊN TẮC TRONG KIỂM THỬ

Để kiểm thử đạt hiệu quả phải tuân thủ các quy tắc sau:

- ✓ Quy tắc 1: Quan trọng ca kiểm thử là định nghĩa đầu ra.
- ✓ Quy tắc 2: Lập trình viên nên tránh tự kiểm tra.
- ✓ Quy tắc 3: Nhóm lập trình không nên kiểm thử chương trình của chính họ.
- ✓ Quy tắc 4: Kiểm tra thấu đáo mọi kết quả.
- ✓ Quy tắc 5: Các ca kiểm thử phải được viết cho các trạng hợp lệ và không hợp lệ

1.2.6 CÁC NGUYÊN TẮC TRONG KIỂM THỬ

Quy tắc 6: Chương trình có thực hiện đúng phần cần thực hiện, phần còn lại liệu chương trình có thực hiện phần không cần thực hiện hay không.

Quy tắc 7: Tránh các ca kiểm thử bâng quơ.

Quy tắc 8: Không dự kiến kết quả kiểm thử theo giả thiết ngầm là không tìm thấy lỗi.

Quy tắc 9: Xác suất tồn tại lỗi trong đoạn chương trình là tương ứng với số lỗi đã tìm thấy trong đoạn đó.

Quy tắc 10: Kiểm thử là nhiệm vụ cực kỳ sáng tạo và có tính thử thách trí tuệ.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Xác minh (Verification)

- ✓ Là quy trình xác định xem sản phẩm có thỏa mãn các yêu cầu đặt ra không.
- ✓ Quan tâm tới việc ngăn chặn lỗi giữa các công đoạn
- ✓ Các hoạt động của xác minh: Kiểm thử (Testing) và Rà soát lại (Review)

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Thẩm định (Validation)

- ✓ Là tiến trình nhằm chỉ ra toàn bộ hệ thống đã phát triển phù hợp với yêu cầu.
- ✓ Là quá trình kiểm chứng xây dựng phầm mềm đúng yêu cầu khách hàng.
- ✓ Chỉ quan tâm đến sản phẩm cuối cùng không còn lỗi.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm thử HỘP ĐEN

- ✓ PPkiểm thử chức năng (functional test) dựa trên đặc tả các chức năng.
- ✓ Phát hiện các sai sót về chức năng (không quan tâm đến cách hiện thực).
- ✓ Do không kiểm thử mọi trường hợp nên sẽ chia nhỏ không gian kiểm thử dựa vào giá trị nhập xuất.
- ✓ Ứng với mỗi vùng dữ liệu, sẽ thiết kế những bộ kiểm thử, đặc biệt tại các giá trị biên.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm thử HỘP ĐEN

- ✓ Với chương trình giải PT bậc 2 theo PP hộp đen

➔Chia không gian kiểm thử thành 4 vùng:

"Vô số nghiệm"

"Vô nghiệm"

"Có nghiệm kép"

"Có 2 nghiệm riêng biệt".

- ✓ Với các bộ kiểm thử đã thiết kế, cần mở rộng:

Biên của số nguyên (32767, -32768)

Số không

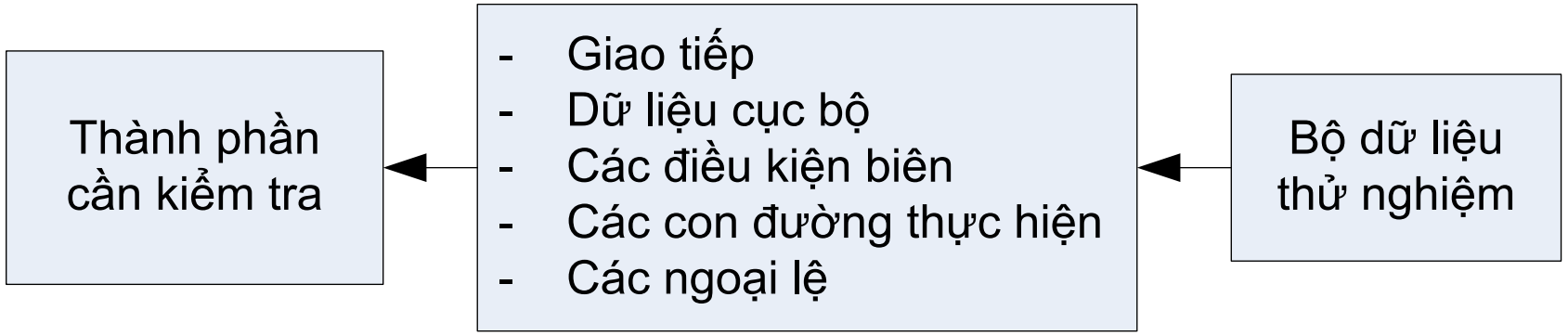
Số âm,

Dữ liệu sai kiểu, dữ liệu ngẫu nhiên….

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm thử HỘP TRẮNG

- ✓Phương pháp hướng đến việc kiểm thử cấu trúc
- ✓Chia không gian kiểm thử dựa vào cấu trúc của đơn vị



1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm thử HỘP TRẮNG

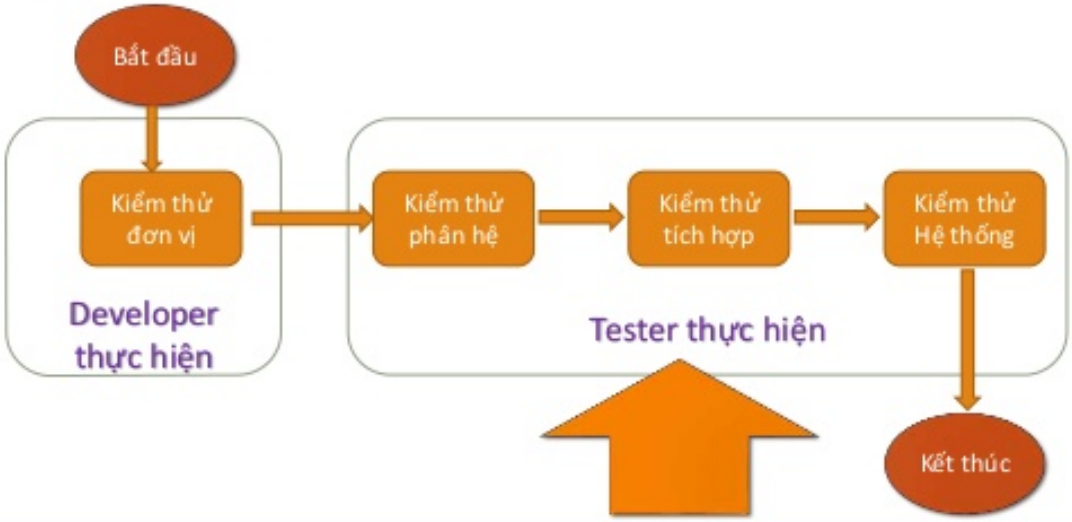
- ✓ *Kiểm tra giao tiếp của đơn vị:* Đảm bảo thông tin vào ra luôn đúng (đúng giá trị, khớp kiểu …)
- ✓ *Kiểm tra dữ liệu cục bộ:* Để đảm bảo dữ liệu được lưu trữ, xử lý oàn vẹn (nhập liệu sai, tên biến sai, kiểu dữ liệu không nhất quán, ...)
- ✓ *Kiểm tra các điều kiện biên:* Các câu lệnh điều kiện/ vòng lặp đảm bảo luôn chạy đúng tại các biên.
- ✓ Kiểm tra các *con đường thực hiện* phải được đi qua ít nhất một lần.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Các giai đoạn kiểm thử:

Với dự án lớn, những người kiểm thử được chia:

- Nhóm 1: kiểm tra các đơn vị đảm bảo thực hiện đúng theo thiết kế.
- Nhóm 2: Các chuyên gia độc lập, có nhiệm vụ phát hiện các lỗi do nhóm thứ 1 chủ quan để lại.



1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm thử đơn vị (unit test)

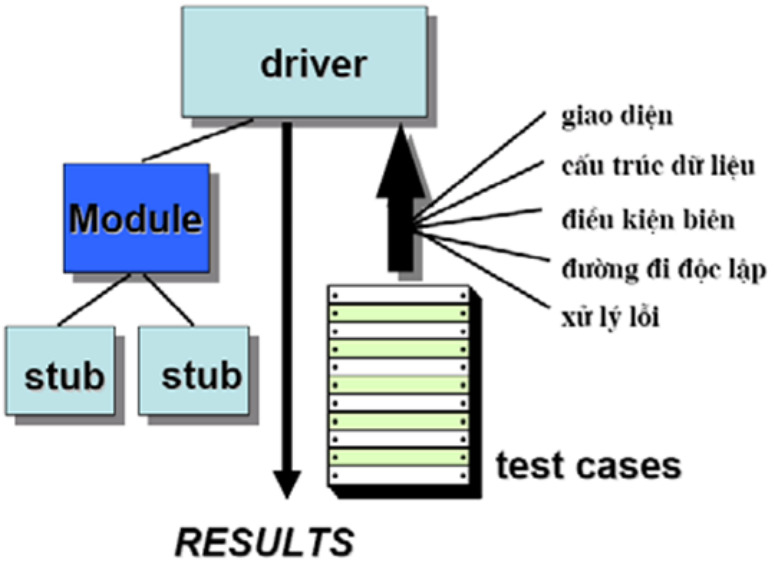
Việc kiểm thử được tiến hành qua các giai đoạn:

- ✓ *Sử dụng kỹ thuật hộp trắng* và dựa vào *hồ sơ thiết kế* để xây dựng các bộ thử nghiệm *sao cho khả năng phát hiện lỗi là lớn nhất*.
- ✓ Vì đơn vị được kiểm tra thường chỉ là một đoạn chương trình hay một thủ tục, nên dù hoàn tất, cũng *không nên giả thuyết tính đúng đắn của các đơn vị* phải xây dựng các mô-đun giả lập đơn vị gọi (*driver*) và đơn vị bị gọi (đặt tên là *stub*):

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm thử đơn vị (unit test)

- ✓ *Driver* như một chương trình chính nhập các bộ kiểm thử và gởi chúng đến đơn vị cần kiểm tra đồng thời nhận kết quả trả về của đơn vị cần kiểm tra.
- ✓ *Stub* là chương trình giả lập thay thế các đơn vị được gọi bởi đơn vị cần kiểm tra. Stub thực hiện các thao tác xử lý dữ liệu đơn giản như: kiểm tra dữ liệu nhập và trả kết quả.



1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm tra tích hợp (Integration Test)

- ✓ Là bước kiểm thử giao diện tích hợp giữa các chức năng khác nhau trong hệ thống.
- ✓ Là bước tiếp theo sau việc kiểm thử từng chức năng.
- ✓ Việc kiểm thử từng chức năng chưa đầy đủ vì có thể riêng rẽ từng chức năng chạy tốt nhưng khi kết hợp lại thì lại không chạy được.
- ✓ Loại kiểm thử này là theo mô hình hộp đen.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm tra hệ thống (System Test)

- ✓ Việc này sẽ thực hiện kiểm thử toàn bộ những chức năng của hệ thống thành một chuỗi thực hiện liên hoàn và kiểm thử một số những chức năng mang tính chất hệ thống (khi tích hợp hai hay nhiều chức năng với nhau thì đặc tính này chưa thể hiện được).
- ✓ Đây là một loại kiểm thử theo mô hình hộp đen.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm tra chấp nhận (Acceptance Test) – Alpha Test

- ✓ Đây là bước cuối cùng trong giai đoạn kiểm thử,
- ✓ Khách hàng sẽ kiểm thử và ký vào biên bản chấp nhận sản phẩm
- ✓ Những tiêu chí chấp nhận đã được thiết lập trong đơn hàng, những điều kiện mà phần mềm cần thỏa mãn để khách hàng chấp nhận sản phẩm.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm tra Beta

- ✓ Là giai đoạn mở rộng của Alpha testing.
- ✓ Khi đó, việc kiểm thử được thực hiện bởi *một số lượng lớn* người sử dụng.
- ✓ Công việc kiểm thử được tiến hành ngẫu nhiên mà không có sự hướng dẫn của các nhà phát triển.
- ✓ Các lỗi nếu được phát hiện sẽ được thông báo lại cho nhà phát triển

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm thử hồi quy (Regression test)

- ✓ Là kiểm thử lại chương trình sau khi thực hiện những thay đổi phần mềm hoặc những thay đổi môi trường.
- ✓ Quá trình này có thể dùng các công cụ tự động thực hiện, hữu ích, đỡ tốn công sức của con người.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm tra tính tương thích (Compatibility Test)

- ✓ Là việc kiểm tra xem hệ thống có tương thích với các môi trường nền khác nhau
- ✓ Như: Kiểm tra xem chương trình có chạy tốt trong các trình duyệt khác nhau không? có chạy được trong hệ điều hành Window /Macintos….

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

- ✓ *Kiểm thử hiệu suất phần mềm*:
  - *Kiểm thử lượng tải:* Kiểm thử hệ thống dưới một lượng tải lớn dữ liệu/lượng lớn người sử dụng(kiểm thử sức chịu đựng).
  - *Kiểm tra khối lượng:* Kiểm tra các chức năng của phần mềm khi một số thành phần tăng triệt để kích thước.
  - *Kiểm thử tính ổn định*: Kiểm tra phần mềm có thể hoạt động tốt liên tục trong một chu kỳ chấp nhận được

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kịch bản kiểm thử

Kịch bản kiểm thử (test script) có hai dạng.

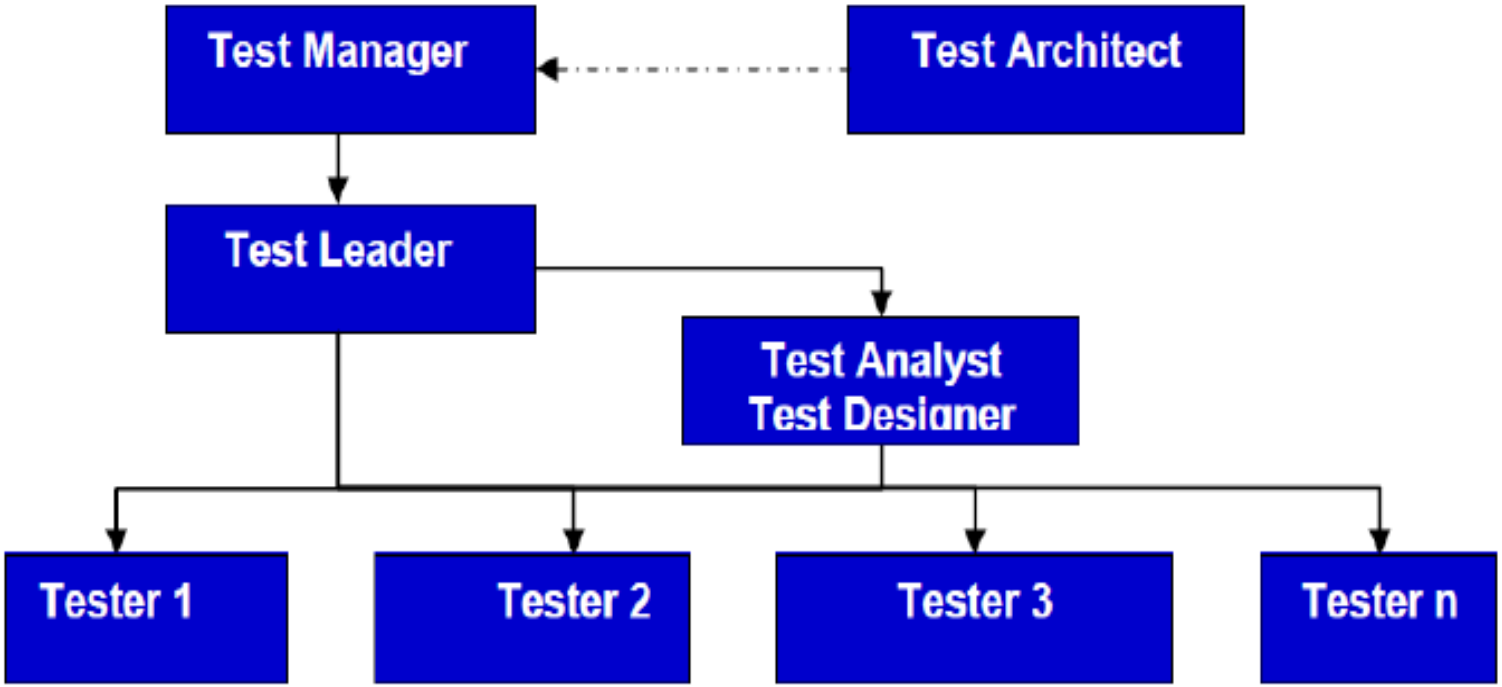
- ✓ Thứ nhất là tập các hướng dẫn thực hiện từng bước với mục đích dẫn dắt nhân viên kiểm thử thực hiện thành công việc kiểm tra phần mềm đó.
- ✓ Thứ hai là một đoạn chương trình nhỏ phục vụ cho việc kiểm thử một cách tự động.

1.2.7 MỘT SỐ KHÁI NIỆM LIÊN QUAN KIỂM THỬ

Kiểm thử tĩnh (Static Testing)

- ✓ Là một phương pháp kiểm tra chéo, người này kiểm tra sản phẩm của của một người khác cùng nhóm.
- ✓ Việc kiểm tra chéo này có tác dụng giảm các lỗi sớm và hiệu quả,

1.2.8 CÁC ĐỐI TƯỢNG THỰC HIỆN KIỂM THỬ

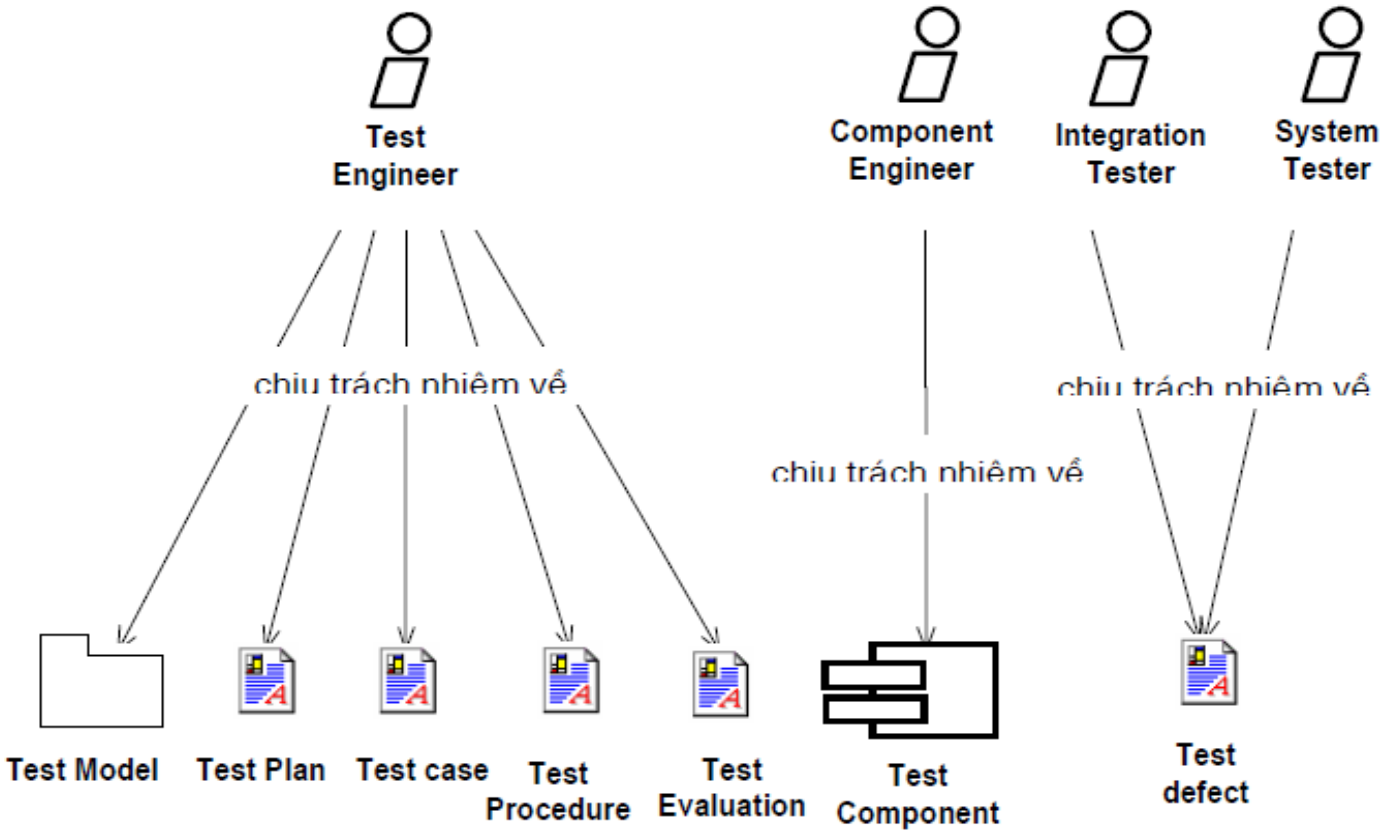


sơ đồ tổ chức kiểm thử

1.2.8 CÁC ĐỐI TƯỢNG THỰC HIỆN KIỂM THỬ

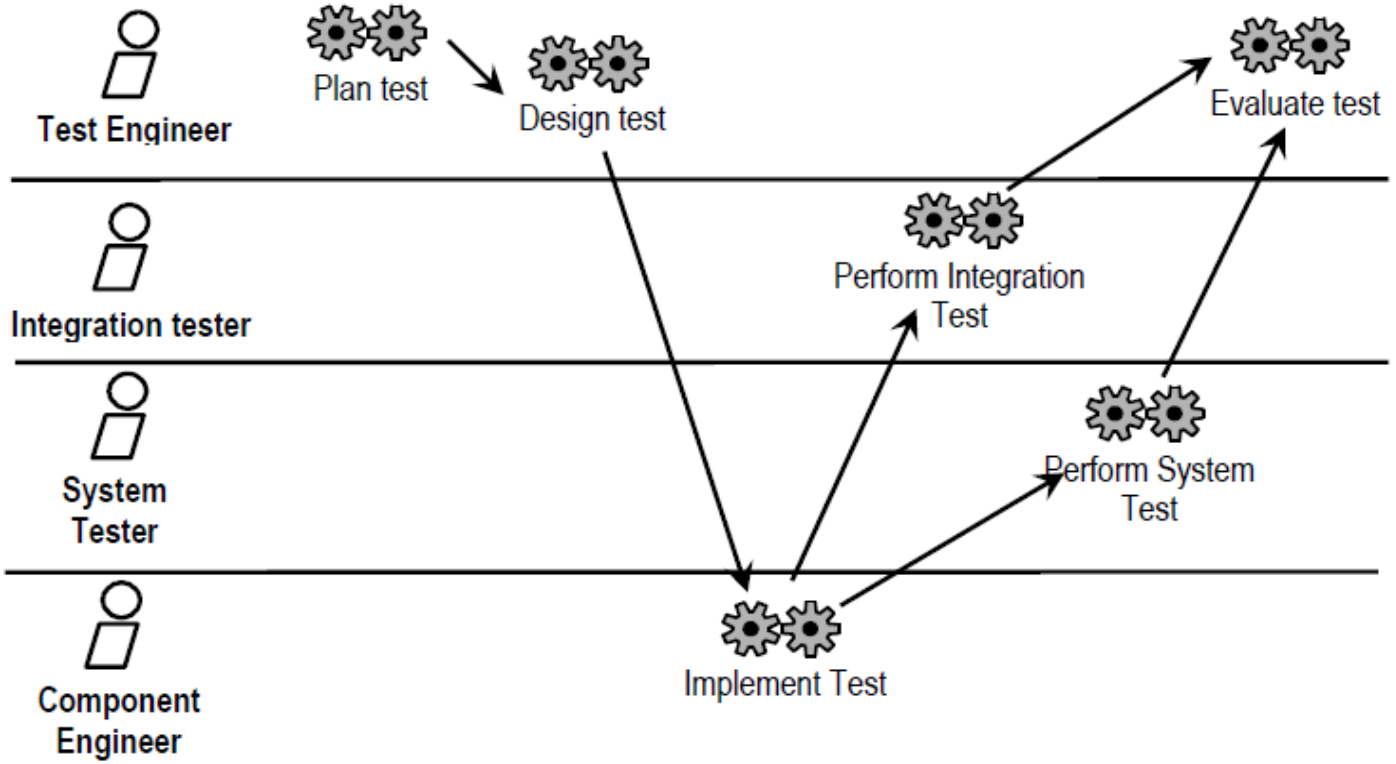
- ✓ *Vị trí nhóm trưởng*: Lập kế hoạch, Tuyển nhân lực, lựa chọn công cụ, quản lý hoạt động của cả đội QA.
- ✓ *Vị trí kỹ sư kiểm thử*: Thực hiện việc kiểm thử, tạo ra các đoạn chương trình để hướng dẫn kiểm thử hoặc thực hiện việc kiểm thử một cách tự động.
- ✓ *Vị trí quản trị hệ thống*: Hỗ trợ cho nhóm QA làm việc (không phải là một thành viên chính thức)
- ✓ *Vị trí biên tập và soạn thảo các tài liệu cho dự án*: Hỗ trợ nhóm QA (không phải là thành viên chính thức).

1.2.8 CÁC ĐỐI TƯỢNG THỰC HIỆN KIỂM THỬ



Worker và các qui trình

1.2.8 CÁC ĐỐI TƯỢNG THỰC HIỆN KIỂM THỬ



Worker và công việc kiểm thử

1.2.9 CÁC ĐIỂM CẦN LƯU Ý KHI KIỂM THỬ

- ✓ Chất lượng không phải do khâu kiểm thử mà do thiết kế quyết định
- ✓ Tính dễ kiểm thử phụ thuộc vào cấu trúc chương trình
- ✓ Người kiểm thử nên làm việc độc lập với người phát triển phần mềm
- ✓ Dữ liệu thử cho kết quả bình thường thì không có ý nghĩa nhiều, cần những dữ liệu thử để phát hiện ra lỗi.
- ✓ Khi phát sinh thêm trường hợp kiểm thử thì nên thử lại những trường hợp trước để tránh lan truyền.

1.2.10 CÁC HẠN CHẾ CỦA KIỂM THỬ

- ✓ Không thể chắc các đặc tả đều đúng 100%.
- ✓ Không thể chắc hệ thống/công cụ kiểm thử là đúng.
- ✓ Không có công cụ kiểm thử nào thích hợp cho mọi PM.
- ✓ Kỹ sư kiểm thử không chắc hiểu đầy đủ về phần mềm.
- ✓ Không có đủ tài nguyên để thực hiệm kiểm thử đầy đủ.
- ✓ không chắc rằng đạt đủ 100% hoạt động

1.3. VAI TRÒ CỦA KIỂM THỬ PHẦN MỀM

KTPM là qui trình chứng minh phần mềm đang có lỗi:

- ✓ Chỉ ra PM hiện đúng các chức năng mong muốn.
- ✓ Là qui trình thiết lập sự tin tưởng về phần mềm.
- ✓ Là qui trình thi hành PM với ý định tìm kiếm các lỗi.
- ✓ Là qui trình tìm kiếm các lỗi của PM để "hủy diệt".

1.3. VAI TRÒ CỦA KIỂM THỬ PHẦN MỀM

Các mục tiêu chính của kiểm thử phần mềm:

- ✓ Phát hiện càng nhiều lỗi càng tốt trong thời gian xác định.
- ✓ Chứng minh phần mềm phù hợp với các yêu cầu.
- ✓ Xác thực chất lượng kiểm thử phần mềm.
- ✓ Tạo các testcase chất lượng, thực hiện kiểm thử hiệu quả và tạo ra các báo cáo đúng.
- ✓ KTPM là một phần trong lĩnh vực rộng hơn, đó là Verification & Validation (V &V).

HUTECH
Đại học Công nghệ Tp.HCM

KIỂM THỬ PHẦN MỀM

Bài 1:
TỔNG QUAN VỀ KIỂM THỬ PHẦN MỀM

Q&A

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

1/. Mục tiêu của công nghệ phần mềm là để:

A/. Thực thi phần cứng tốt hơn

B/. Chỉnh sửa lỗi phần mềm

C/. Có thể sử dụng lại phần mềm

D/. Tạo sản phẩm phần mềm chất lượng hơn

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

2/. Theo thống kê từ những thách thức đối với CNPM thì lỗi nhiều nhất là do:

A/. Kiểm tra và bảo trì

B/. Thiết kế

C/. Lập trình

D/. Phân tích yêu cầu

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

3/. Kỹ sư phần mềm không cần

A/. Kiến thức về phân tích thiết kế hệ thống.

B/. Kiến thức về cơ sở dữ liệu.

C/. Lập trình thành thạo bằng một ngôn ngữ lập trình.

D/. Kinh nghiệm quản lý dự án phần mềm.

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

4/. SDLC là viết tắt của:

A. Spiral Development Linear Cycle

B. Software Development Life Cycle

C. System Development Line Cycle

D. Sequential Development Linear Cycle

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

5/. Nếu yêu cầu là dễ hiểu và xác định được thì mô hình nào là thích hợp nhất để phát triển hệ thống:

A. Mô hình thác nước (waterfall model)

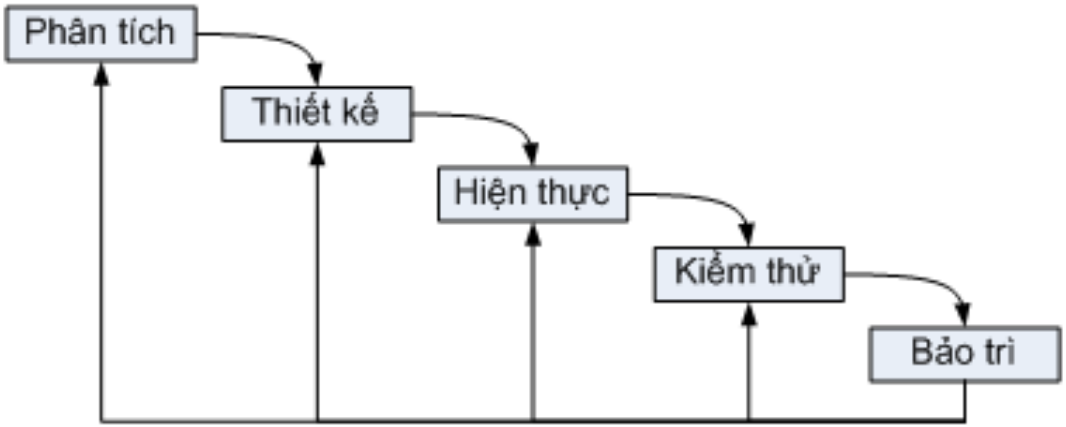
B. Mô hình bản mẫu (prototyping)

C. Mô hình xoắn ốc (spiral model)

D. Mô hình phát triển nhanh (RAD)

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

6/. Hình ảnh minh họa sau thể hiện các hoạt động của mô hình phát triển phần mềm nào?



A. Mô hình bản mẫu(prototyping)

B. Mô hình thác nước (waterfall model)

C. Mô hình xoắn ốc (spiral model)

D. Mô hình phát triển nhanh (RAD)

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

7/. Mô hình prototyping là:

A. Mô hình rất thích hợp khi các yêu cầu hệ thống được xác định rõ ràng

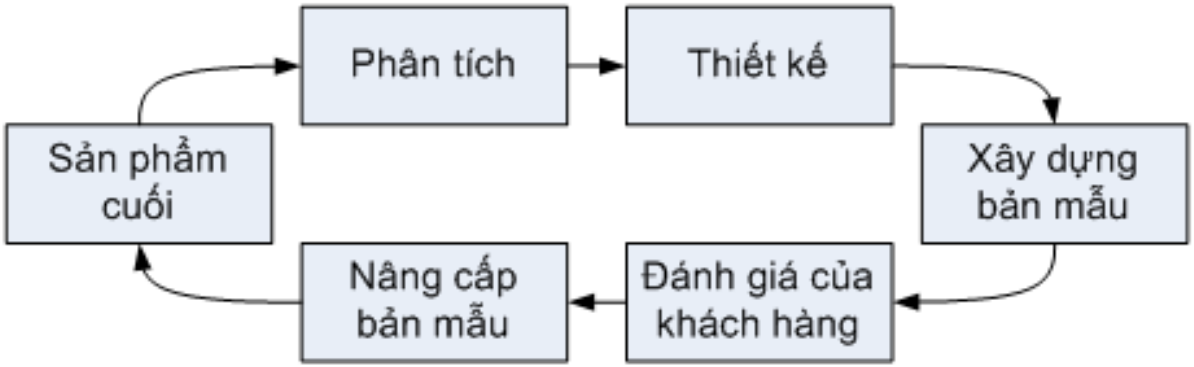
B. Mô hình thường dùng khi khách hàng không thể xác định được yêu cầu rõ ràng

C. Mô hỉnh tốt nhất cho những dự án có nhiều đội phát triển cùng tham gia

D. Mô hình nhiều rủi ro nên ít khi tạo ra sản phẩm có giá trị

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

8/. Hình ảnh minh họa sau thể hiện các hoạt động của mô hình phát triển phần mềm nào?



A. Mô hình bản mẫu(prototyping)

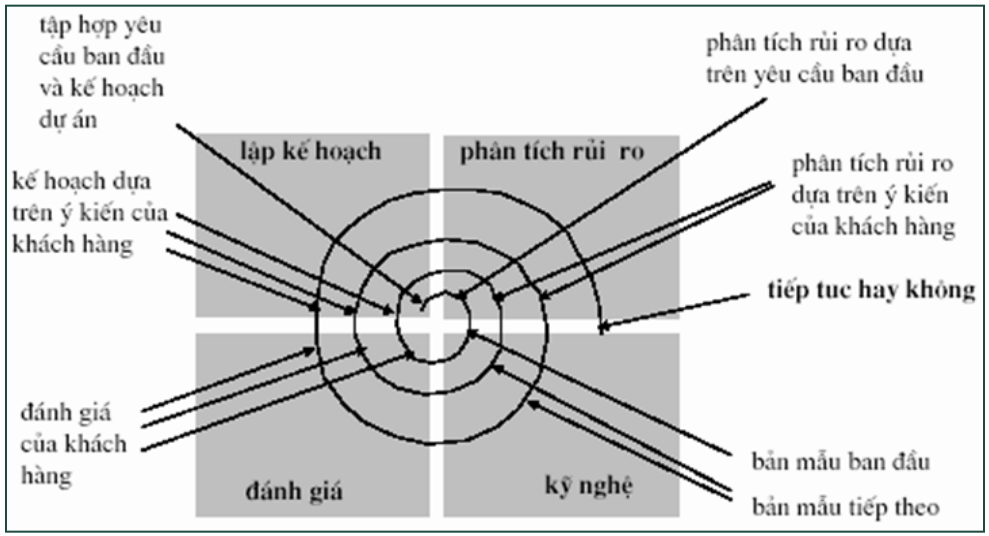
B. Mô hình thác nước (waterfall model)

C. Mô hình xoắn ốc (spiral model)

D. Mô hình phát triển nhanh (RAD)

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

9/. Hình ảnh minh họa sau thể hiện các hoạt động của mô hình phát triển phần mềm nào?



A. Mô hình bản mẫu(prototyping)

B. Mô hình thác nước (waterfall model)

C. Mô hình xoắn ốc (spiral model)

D. Mô hình phát triển nhanh (RAD)

TRẮC NGHIỆM VỀ QUY TRÌNH PHÁT TRIỂN PM

10/. Mô hình phát triển phần mềm mà tiến trình tiến hoá vốn cặp đôi bản chất lặp của làm bản mẫu với các khía cạnh hệ thống và có kiểm soát của mô hình trình tự tuyến tính. Là mô hình phát triển phần mềm nào sau đây?

A. Mô hình tăng trưởng (incremental model)

B. Mô hình kĩ thuật thế hệ thứ tư (Fourth generation techniques - 4GT)

C. Mô hình xoắn ốc (spiral model)

D. Mô hình RAD (Rapid application development)

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

1/. Kiểm thử phần mềm là:

A/. Quá trình nhằm chứng minh là phần mềm không có lỗi.

B/. Quá trình nhằm xác lập độ tin cậy vào chương trình.

C/. Quá trình thực thi chương trình để chỉ ra là nó làm việc theo đúng đặc tả hệ thống.

D/. Quá trình thực thi chương trình để cố gắng tìm ra lỗi.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

2/. STLC là viết tắt của từ

A/. System Testing Life Cycle.

B/. Software Testing Life Cycle.

C/. Software Test Life Cycle.

D/. System Test Life Cycle.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

3/. Verification là:

A/. Kiểm chứng xem sản phẩm có đáp ứng với mong đợi của khách hàng không?

B/. Kiểm chứng xem sản phẩm có đúng với mong đợi của nhà phát triển không?

C/. Kiểm chứng xem sản phẩm có đáp ứng với các ràng buộc của dự án không?

D/. Kiểm chứng xem sản phẩm có phù hợp với môi trường hệ điều hành không?

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

4/. Validation là:

A/. Xác nhận xem sản phẩm có đáp ứng với yêu cầu của khách hàng không?

B/. Xác nhận xem sản phẩm có đúng với yêu cầu của nhà phát triển không?

C/. Xác nhận xem sản phẩm có đáp ứng với các ràng buộc của dự án không?

D/. Xác nhận xem sản phẩm có tương thích với môi trường hệ điều hành không?

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

5/. Tester cần tố chất nào sau đây:

A/. Có óc sáng tạo.

B/. Giỏi về lập trình.

C/. Giỏi về phân tích, thiết kế hệ thống.

D/. Có óc phán đoán tốt.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

6/. Kiểm thử phần mềm hướng đến

A/. Phát hiện tất cả các lỗi.

B/. Sửa các lỗi tìm thấy.

C/. Ngăn ngừa và đề phòng lỗi xảy ra.

D/. Bảo đảm phần mềm hoàn toàn sạch lỗi.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

7/. Quy trình kiểm thử theo mô hình V gồm các giai đoạn sau:

A/. Requirements analysis, Test planning, Test execution, Test reporting, Test result analysis.

B/. Requirements analysis, Test planning, Test execution, Test reporting, Test result analysis, Defect Retesting, Regression testing, Test closure.

C/. Requirements analysis, Test execution, Test reporting, Test result analysis, Defect Retesting, Regression testing, Test closure.

D/. Test planning, Test execution, Test reporting, Test result analysis, Defect Retesting, Regression testing, Test closure.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

8/. Kiểm thử tĩnh là:

A/. Là kiểm thử không cần đến máy tính để chạy chương trình, nó chỉ rà soát mã lệnh, kiểm tra và duyệt qua các yêu cầu phần mềm.

B/. Là kiểm thử dựa trên đầu vào và đầu ra của chương trình mà không quan tâm tới mã lệnh bên trong.

C/. Là kiểm thử dựa vào thuật toán, cấu trúc dữ liệu và mã lệnh bên trong của chương trình.

D/. Là kiểm thử có chạy chương trình với số liệu mẫu ít nhất 1 lần.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

9/. Kiểm thử động là:

A/. Là kiểm thử có chạy chương trình để kiểm tra khả năng xử lý dữ liệu lỗi.

B/. Là kiểm thử có chạy chương trình nhiều lần với nhiều giá trị đầu vào khác nhau.

C/. Là kiểm thử có chạy chương trình với các giá trị đầu vào ít nhất 1 lần.

D/. Là kiểm thử liên quan đến chạy chương trình với các giá trị đầu vào và kiểm tra đầu ra có được như mong đợi với các Test case cụ thể không.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

10/. QA là viết tắt của từ:

A/. Quality Assure.

B/. Quality Assurance.

C/. Quantity Assurance.

D/. Quality Assure.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

11/. QC là viết tắt của từ:

A/. Quality Control.

B/. Quantity Computer.

C/. Quantity Control.

D/. Quality Computer.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

12/. Black box testing là:

A/. Là phương pháp kiểm thử dựa trên đầu vào, đầu ra không quan tâm tới mã lệnh của chương trình.

B/. Là phương pháp kiểm thử dựa vào thuật toán, cấu trúc dữ liệu và mã lệnh bên trong của chương trình.

C/. Là phương pháp kiểm thử dựa trên đầu vào, đầu ra và mã lệnh bên trong chương trình.

D/. Là phương pháp kiểm thử dựa trên đầu vào, đầu ra ứng với các vùng giá trị.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

13/. White box testing là:

A/. Là phương pháp kiểm thử dựa trên đầu vào, đầu ra không quan tâm tới mã lệnh của chương trình.

B/. Là phương pháp kiểm thử dựa vào thuật toán, cấu trúc dữ liệu và mã lệnh của chương trình.

C/. Là phương pháp kiểm thử dựa trên đầu vào, đầu ra và mã lệnh của chương trình.

D/. Là phương pháp kiểm thử dựa trên đầu vào, đầu ra ứng với các giá trị biên.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

14/. Grey box testing là:

A/. Là phương pháp kiểm thử dựa trên đầu vào, đầu ra không quan tâm tới mã lệnh của chương trình.

B/. Là phương pháp kiểm thử dựa vào thuật toán, cấu trúc dữ liệu và mã lệnh của chương trình.

C/. Là phương pháp kết hợp giữa Black box testing và White box testing.

D/. Là phương pháp kiểm thử thủ công dựa trên khả năng đoán lỗi của Tester.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

15/. Functional testing là:

A/. Kiểm thử các hàm của chương trình có thực hiện đúng không?

B/. Kiểm thử các hàm của chương trình có bị lỗi không?

C/. Kiểm thử các chức năng cơ bản, quy trình nghiệp vụ có đáp ứng yêu cầu của người sử dụng không.

D/. Kiểm thử để chấp nhận sản phẩm.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

16/. Unit testing là:

A/. Kiểm thử các đơn vị chương trình như:

Function, Procedure, Class, Method,...

B/. Kiểm thử các Module chương trình.

C/. Kiểm thử tích hợp các đơn vị chương trình.

D/. Kiểm thử toàn bộ hệ thống.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

17/. Unit testing được thực hiện bởi:

A/. Phân tích viên.

B/. Nhân viên kiểm thử.

C/. Kỹ sư công nghệ.

D/. Lập trình viên.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

18/. Alpha test là:

A/. Kiểm thử trước khi xuất ra thị trường với phiên bản đầu tiên.

B/. Kiểm thử ngay tại nơi phát triển phần mềm bởi 1 nhóm kiểm thử độc lập.

C/. Kiểm thử sau khi phát hiện các lỗi và đã sửa lỗi.

D/. Kiểm thử tích hợp bởi 1 nhóm kiểm thử độc lập.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

19/. Beta test là:

A/. Kiểm thử trước khi xuất ra thị trường với phiên bản thứ 2, 3.

B/. Kiểm thử sau khi phát hiện các lỗi và đã sửa lỗi.

C/. Kiểm thử tích hợp bởi 1 nhóm kiểm thử độc lập.

D/. Kiểm thử bởi người sử dụng tiềm năng trong môi trường thực tế, và mọi phản hồi sẽ gửi trả về nhà phát triển phần mềm.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

20/. Reliability testing là:

A/. Kiểm thử hiệu quả.

B/. Kiểm thử độ tin cậy.

C/. Kiểm thử khả năng bảo trì.

D/. Kiểm thử an ninh, bảo mật.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

21/. Usability testing là:

A/. Kiểm thử khả năng sử dụng.

B/. Kiểm thử độ tin cậy.

C/. Kiểm thử khả năng bảo trì.

D/. Kiểm thử khả năng tương thích.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

22/. Efficiency testing là:

A/. Kiểm thử khả năng tương thích.

B/. Kiểm thử hiệu quả.

C/. Kiểm thử chịu tải.

D/. Kiểm thử độ tin cậy.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

23/. Maintainability testing là:

A/. Kiểm thử khả năng bảo trì.

B/. Kiểm thử cơ sở.

C/. Kiểm thử tài liệu.

D/. Kiểm thử quá tải.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

24/. Compatibility testing là:

A/. Kiểm thử quá tải.

B/. Kiểm thử khả năng tương thích.

C/. Kiểm thử khả năng bảo trì.

D/. Kiểm thử độ tin cậy.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

25/. Documentation testing là:

A/. Kiểm thử khả năng sử dụng.

B/. Kiểm thử khả năng tương thích.

C/. Kiểm thử tài liệu.

D/. Kiểm thử sức chịu đựng.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

26/. Endurance testing là:

A/. Kiểm thử sức chịu đựng.

B/. Kiểm thử khả năng sử dụng.

C/. Kiểm thử quá tải.

D/. Kiểm thử chịu tải.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

27/. Load testing là:

A/. Kiểm thử khả năng sử dụng.

B/. Kiểm thử quá tải.

C/. Kiểm thử chịu tải.

D/. Kiểm thử an ninh, bảo mật.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

28/. Performance testing:

A/. Kiểm thử sức chịu đựng.

B/. Kiểm thử quá tải.

C/. Kiểm thử khả năng sử dụng.

D/. Kiểm thử hiệu suất.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

29/. Security testing là:

A/. Kiểm thử an ninh, bảo mật.

B/. Kiểm thử tài liệu.

C/. Kiểm thử hiệu quả.

D/. Kiểm thử khả năng sử dụng.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

30/. Scalability testing là:

A/. Kiểm thử khả năng mở rộng.

B/. Kiểm thử quá tải.

C/. Kiểm thử chịu tải.

D/. Kiểm thử độ tin cậy.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

31/. Volume testing là :

A/. Kiểm thử quá tải.

B/. Kiểm thử hiệu quả.

C/. Kiểm thử khối lượng dữ liệu.

D/. Kiểm thử độ tin cậy.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

32/. Stress testing là:

A/. Kiểm thử chịu tải.

B/. Kiểm thử quá tải.

C/. Kiểm thử hiệu quả.

D/. Kiểm thử khả năng sử dụng.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

33/. Recovery testing là:

A/. Kiểm thử quá tải.

B/. Kiểm thử khả năng sử dụng.

C/. Kiểm thử phục hồi.

D/. Kiểm thử khả năng mở rộng.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

34/. Kiểm thử cấu trúc thường (Structural testing) là 1 hình thức khác của:

A/. White box testing.

B/. Black box testing.

C/. Unit testing.

D/. Grey box testing.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

35/. Automated testing là:

A/. Kiểm thử thủ công bằng cách đoán lỗi.

B/. Sử dụng phần mềm đặc biệt nhằm tự động thực hiện các Test case và so sánh kết quả thực tế với kết quả dự đoán.

C/. Kiểm thử tập trung vào các chức năng quan trọng của hệ thống và không quan tâm đến kiểm tra chi tiết.

D/. Kiểm thử tuân thủ các tiêu chuẩn quốc tế.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

36/. Kiểm thử hộp kính (Glass box testing) còn được là:

A/. Black box testing.

B/. Unit testing.

C/. Grey box testing.

D/. White box testing.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

37/. Integration testing là:

A/. Là Unit testing.

B/. Là Volume testing.

C/. Là kiểm thử tích hợp, nó kết hợp các thành phần của phần mềm và kiểm thử như một ứng dụng đã hoàn thành.

D/. Là Kiểm thử toàn bộ hệ thống.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

38/. System testing là:

A/. Kiểm thử phần cứng và phần mềm của hệ thống.

B/. Kiểm thử tích hợp lần cuối trước khi phát hành.

C/. Kiểm thử sau khi có 1 thay đổi được cập nhật.

D/. Kiểm tra các thiết kế và toàn bộ hệ thống (sau khi đã kiểm thử tích hợp) có thỏa mãn yêu cầu đặt ra hay không?

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

39/. Chọn phát biểu đúng?

A/. Mô hình kiểm thử phần mềm độc lập với mô hình phát triển phần mềm.

B/. Mô hình kiểm thử phần mềm phụ thuộc vào mô hình phát triển phần mềm.

C/. Chỉ có một mô hình kiểm thử phần mềm chữ V.

D/. Không cần thiết có mô hình kiểm thử phần mềm.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

40/. Kiểm thử hồi quy (Regression testing) thường có liên quan nhiều đến:

A/. Kiểm thử chức năng.

B/. Kiểm thử dòng dữ liệu.

C/. Kiểm thử khi phát triển hệ thống (Development testing).

D/. Kiểm thử trong lúc bảo trì (Maintenance testing)

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

41/. Kiểm thử mã lệnh là

A/. Black box testing.

B/. Red box testing.

C/. White box testing.

D/. Grey box testing.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

42/. Acceptance testing được hiểu là:

A/. Grey box testing.

B/. White box testing.

C/. Alpha Testing.

D/. Beta testing.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

43/. Mục nào là kiểm thử phi chức năng (non-functional testing)?

A/. Black box testing.

B/. Performance testing.

C/. Unit testing.

D/. Alpha Testing.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

44/. Behavioral testing là:

A/. White box testing.

B/. Black box testing.

C/. Grey box testing.

D/. Red box testing.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

45/. Kiểm thử phần mềm với số liệu thật trong môi trường thực tế gọi là:

A/. Alpha testing.

B/. Beta testing.

C/. Regression testing.

D/. Volume testing.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

46/. Alpha testing được hoàn thành bởi

A/. Users

B/. Developers

C/. Testers

D/. Tất cả các bên có liên quan.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

47/. Beta testing là

A/. Một hình thức mở rộng của kiểm thử mức chấp nhận của người dùng.

B/. Kiểm thử hiệu suất phần mềm.

C/. Kiểm thử xem giao diện có tiện dụng và dễ hiểu với người dùng không.

D/. Kiểm thử sự phá hủy, bằng cách cố gắng làm hỏng phần mềm hoặc một hệ thống con.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

48/. Để kiểm tra việc phát triển 1 phần mềm thỏa có phù hợp với yêu cầu của khách hàng hay không, đó là các quá trình nào?

A/. Verification, Validation.

B/. Validation , Verification.

C/. Quality Assurance, Verification.

D/. Quality Control, Verification.

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

49/. Thử nghiệm nào duyệt qua các mã lệnh của phần mềm:

A/. Unit testing.

B/. Black box testing.

C/. White box testing.

D/. Regression testing

TRẮC NGHIỆM VỀ KIỂM THỬ PHẦN MỀM

50/. Mật độ lỗi (defect density) dùng để đo lường

A/. Chất lượng sản phẩm cuối.

B/. Dự án phần mềm.

C/. Độ phức tạp của code.

D/. Chất lượng bảo trì.