**Câu 1 : Alpha release là gì? Khác gì với beta release?**

Alpha release là phiên bản được kiểm thử hoạt động chức năng thực tế hoặc giả lập do một số ít người dùng/khách hàng được chỉ định hoặc một nhóm test thực hiện tại nơi sản xuất phần mềm. Alpha release được thực hiện ngay sau kiểm thử hệ thống (Product Testing). Alpha release thường áp dụng cho các sản phẩm COTS (MS Office, Windows,..) là một hình thức kiểm thử chấp nhận nội bộ, trước khi phần mềm được tiến hành kiểm thử beta.

**Câu 2 : Beta release là gì? Khác gì với alpha release?**

Được thực hiện sau alpha release, được phát hành tới một số nhóm khách hàng bên ngoài nhóm phát triển phần mềm để tăng phạm vi phản hồi từ người sử dụng tương lai lớn nhất. Beta release gần như phiên bản cuối cùng (final version)

**Câu 3 : Luật Dijkstra trong CNPM nói gì ?**

Luật Dijkstra : Rất dễ để chứng minh một phần mềm có lỗi, nhưng không thể chứng minh được một phần mềm không có lỗi.

**Câu 4 : . Thẩm định và xác minh là gì (V&V)? Tầm quan trọng của chúng?**

Verification & Validation:

- Thẩm định là việc xác định xem luồng công việc (workflow) đã được thực hiện đúng hay chưa, và việc thẩm định sẽ được thực hiện ở cuối mỗi Workflow.

- Xác minh là 1 quy trình đánh giá chuyên sâu được thực hiện trước khi đưa sản phẩm tới tay khách hàng.

- Mục đích của việc này là xác định xem sản phẩm đã thỏa mãn hoàn toàn bản đặc tả chưa

**Câu 5 : Có những loại V&V nào? Mô tả nội dung mỗi loại**

Vì V&V thực hiện thẩm định vào cuối mỗi WorkFlow, trong khi yêu cầu kiểm tra được thực hiện thường xuyên trong quá trình phát triển sản phẩm và hoạt động bảo trì là cần thiết. Cho nên người ta thường sử dụng “testing” thay cho V&V.

- Có 2 loại testing là : execution-based testing(test thực thi) and non-execution based testing(test không cần thực thi).

- execution-based testing: Sử dụng đầu vào định sẵn, với nền tảng, môi trường định sẵn. Thực hiện suy diễn từ đầu vào ra kết quả thực thi. (vd: viết test - case chạy code)

- non-execution based testing: kiểm tra phần mềm không sử dụng test-case. (vd: đọc lại source code phần mềm, phân tích logic toán học trong chương trình ). Gồm có 2 loại là WalkThrough và Inspection.

**Câu 6 : inspection là gì? Khác gì với walkthrough?**

- Inspection: Team Inspection thường gồm 4 người. Vd cho Inspect design gồm có: Moderator, designer, implementer, tester. là 1 quy trình gồm 5 bước:

1/ (Overview)Nhìn lại tổng quan toàn bộ tài liệu điều tra (lấy yêu cầu, đặc tả, thiết kế, code hoặc kế hoạch), tài liệu được phát cho các thành viên.

2/ (Preparation) các thành viên hiểu chi tiết tài liệu. Liệt kê danh sách các loại lỗi, sắp xếp theo tần suất xuất hiện. Việc này giúp mọi người tập trung vùng có nhiều lỗi.

3/ inspection 1 người sẽ lướt qua tài liệu, và các thành viên điều tra sẽ xác định lỗi mà không sửa chúng. Và moderator của inspection team trong 1 ngày sẽ có báo cáo, bảo đảm quy trình được thực hiện đúng.

4/ (Rework) các thành viên sửa toàn bộ lỗi được chỉ ra trong báo cáo.

5/ (Follow Up) Moderator phải kiểm tra toàn bộ, bảo đảm không còn lỗi trong báo cáo và không có lỗi phát sinh. Nếu 5% tài liệu trên phải rework (bước 4) lại thì team phải điều tra lại (reinspection).

- Điểm khác so vs Walkthrough: 83 + 84;

**Câu 7 : walkthrough là gì? Khác gì với inspection?**

- Walkthrough: Gồm có 4-6 người, 1 người viết đặc tả (specification), 1 người quản lý workflow phân tích, đại diện khách hàng,1 người thuộc team thiết kế, 1 đại diện thuộc SQA (leader). Có 2 kiểu walkthrough : Document Driven và participants Driven

+ Document Driven: 1 người (presenter) sẽ kiểm tra toàn bộ nội dung, và những người còn lại có trách nhiệm bình luận, ngắt quãng những chỗ nghi ngờ có lỗi. (cách hiệu quả hơn để tìm lỗi)

+Participant Driven: mỗi thành viên chuẩn bị 1 danh sách gồm 2 cột: các chi tiết chưa rõ ràng hoặc chi tiết nghi ngờ dính lỗi. Đại diện team phân tích phải phân tích được đâu là lỗi và làm rõ cho người hỏi.

- Điểm khác so vs Inspection (83 + 84)

**Câu 8** : **Tại sao trong nhóm walkthrough và inspection, luôn phải có đại diện của workflow tiếp theo?**

- Bởi vì người đại diện cho nhóm WF tiếp theo sẽ chuyển những thông tin công việc từ WF trước về thông tin công việc của mình.

**Câu 9** : **Người ta nói « nhóm SQA tạo ra chất lượng cho phần mềm » đúng hay sai? Tại sao?**

- Sai, SQA không tạo ra chất lượng phần mềm, họ chỉ bảo đảm chất lượng phần mềm. Còn người tạo nên chất lượng phần mềm là Developer.(câu này thầy Hùng đã khẳng định)

**Câu 10** : **Phần mềm còn lỗi hay không khi thực hiện hết các ca kiểm thử được thiết kế? Khi nào dừng kiểm thử?**

Bất kỳchuẩn chất lượng mong muốn nào mà phần mềm phải thỏa mãn hầu sẵn sàng cho việc phân phối đến khách hàng. Có thể bao gồm các thứ sau : Các yêu cầu mà phần mềm phải được kiểm thửdưới các môi trường xáx định. Số lỗi tối thiểu ởcấp an ninh và ưu tiên khác nhau, số phủ kiểm thử tổi thiểu,... Stakeholder sign-off and consensus

**Câu 11** **: Trình bày phương pháp kiểm thử hộp trắng: cơ sở phương pháp; các yêu cầu cần kiểm tra, các kỹ thuật được sử dụng.**

- Kiểm thử hộp trắng (test to code, glass-box, white-box, structural, logic-driven, and path-oriented testing) là phương pháp chọn testcase dựa trên việc kiểm tra code, không phải kiểm tra bản đặc tả.

- Kỹ thuật được sử dụng:

+ Statement Coverage: Phương pháp này chạy 1 loạt các testcase song song với việc đảm bảo mỗi câu (statement) được chạy ít nhất 1 lần. Yêu cầu cần có 1 CASE tool để theo dõi câu lệnh nào đó được thực hiện bao nhiêu lần trong test. Điểm yếu của phương pháp này: nó không bảo đảm đầu ra của mọi nhánh (branch) đều được kiểm tra.

+ Branch Coverage: Chạy 1 loạt các test để bảo đảm tất cả các nhánh đều được kiểm tra ít nhất 1 lần. Và cần có 1 tool để biết được nhánh nào đã kiểm tra (hoặc chưa).

+ Path Coverage: Phương pháp hiệu quả nhất, cho phép kiểm tra tất cả các đường đi (path) của chương trình. Tuy nhiên (giả sử trong 1 chương trình có nhiều vòng lặp) thì số đường đi là rất lớn. Do vậy, vấn đề giảm bớt số đường đi và phát hiện đủ lỗi đang được nghiên cứu.

**Câu 12 Trình bày phương pháp kiểm thử hộp đen: cơ sở phương pháp; các yêu cầu cần kiểm tra, các kỹ thuật được sử dụng.**

- Kiểm thử hộp đen ( test to specifications, black-box, behavioral, datadriven, functional, and input/output-driven testing.) là phương pháp chọn test case dựa trên tài liệu đặc tả và bỏ qua code.

- kỹ thuật được sử dụng:

+ Equivalence Testing and Boundary Value Analysis:

+ Equivalent Class (lớp tương đương): (pg521): Tất cả các testcase trong lớp tương đương đều có độ tốt như nhau. Do đó với mỗi lớp tương đương chỉ cần 1 testcase là đủ. VD: quản lý hàng hóa có STT từ 1 đến 2000, ta có 3 khoản Equi Class: (1000) và mỗi khoảng chỉ cần 1 testcase.

+ Boundary Value Analysis (phân tích giá trị biên): Để mở rộng khả năng tìm lỗi, hỗ trợ cho Equivalent Class. vd: một tham số đầu vào có một giới hạn biên x, thì phải test ít nhất 4 trường hợp: • 1: giá trị đầu vào bất kì cách xa x • 2: giá trị đầu vào ngay trên x • 3: giá trị đầu vào ngay dưới x • 4: giá trị đầu vào đúng bằng x

+ Functional Testing: Những dữ liệu kiểm tra sẽ phụ thuộc vào chức năng của code artifact. Sau khi xác định được chức năng của code artifact (vd: code artifact có chức năng xác định có phải số nguyên tố hay không?), tập dữ liệu kiểm tra được sử dụng để kiểm tra các chức năng 1 cách độc lập nhau.

**Câu 13** : **Kiểm thử đơn vị đối tượng là gì? Ai thực hiện. Các phương pháp và kỹ thuật nào được sử dụng? Kiểm tra những loại lỗi nào?**

-Kiểm thử đơn vị (Unit testing) là việc kiểm thử của lập trình viên sau khi hoàn thành xong 1 code. Sau khi kiểm thử đơn vị hoàn tất, code được gửi cho SQA để tiếp tục kiểm tra.

Hàm , lớp phương thức

- Có 2 kỹ thuật cho phép kiểm tra: Hộp trắng & Hộp đen. Như đã mô tả ở trên.

**Câu 14: Các chiến lược nào sử dụng trong kiểm thử tích hợp? Ưu điểm và hạn chế mỗi loại?**

- Top-down thực hiện kiểm thử top layer đầu tiên.

Ưu : thuận theo trình độ chức năng

Nhược : viết stup khá khó

- Bottom – up

Ưu : ko tốn quá nhiều stup

Nhược : chức năng phụ kiểm trước , chức năng chính kiểm thử sau

Sandwich

Ưu :kết hợp 2 loại trên

- Kiểm tra code artifact sử dụng unit testing , sau đó kiểm tra xem sau khi tích hợp vào, sản phẩm mới có bị lỗi/sai sót hay không?

**Câu 15. Giải thích khái nhiệm stub và driver? Chúng được sử dụng ở đâu và vì sao?**

- Stub: Trong quá trình tích hợp các artifact với nhau, có những artifact(giả sử A) cần những artfact khác (vd: B,C,D) để có thể chạy được. Khi đó, B,C,D được gọi là Stub.

- Driver: Trong ví dụ trên, để kiểm tra artifact A ta cần 1 Driver và 3 Stub là B,C,D. (Giả sử) nếu B có thể tự chạy được mà k cần artifact nào, Có thể kiểm tra B với 1 Driver và 0 Stub.

- Chúng được sử dụng trong quá trình tích hợp các artifact với nhau, quá trình test tích hợp để tạo ra sản phẩm hoàn chỉnh.

**Câu 16 . Kiểm thử hệ thống nhằm kiểm tra cái gì? Ai thực hiện? Các phương pháp?**

- (Product Testing): Diễn ra sau khi quá trình kiểm tra tích hợp được hoàn tất. Mục đích bảo đảm phần mềm hoàn tất. Thực hiện bởi SQA Group. Được thực hiện trên 2 loại phần mềm Cost Of The Shelf software và Custom Software.

- Với COTS Software, sẽ có 2 giai đoạn là alpha và beta. nhiều khách hàng sẽ được chọn để thử nghiệm và gửi feedback lại.

- Với Custom Software, SQL Group phải bảo đảm rằng Acceptance Test (bảo đảm đúng theo đặc tả) phải hiệu quả. Nếu không developer sẽ bị mất uy tín.

- Kiểm tra bao gồm các phương pháp:

+ Black Box

+ Robustness of Software.

+ Phải thỏa mãn tất cả các ràng buộc của phần mềm.

+ Kiểm tra toàn bộ code

+ tài liệu gửi đến cho khách hàng.

**Câu 17 . Trình bày các kiểm thử được thực hiện trong kiểm thử hệ thống?**

+ Black Box

+ Robustness of Software.

+ Phải thỏa mãn tất cả các ràng buộc của phần mềm.

+ Kiểm tra toàn bộ code

+ tài liệu gửi đến cho khách hàng.

**Câu 18. Kiểm thử chấp nhận là gì? Trong đó có những kiểm thử nào được thực hiện? Phân biệt**.

(Acceptance Test) là kiểm tra xem phần mềm đã hoạt động đúng như đặc tả đưa ra bởi developer hay chưa. Acceptance Test được thực hiện trên dữ liệu thực tế. Được thực hiện bởi 1 trong 3 nhóm đối tượng: đại diện khách hàng và nhóm SQA, hoặc nhóm SQA do khách hàng thuê.

- Có các kiểu kiểm thử sau:

+ Kiểm thử đúng (Correctness testing)

+ Hiệu năng (Performance) (tùy chọn)

+ Robustness

+ Tài liệu (Documentation