Lý thuyết
Kiểm Tra Phần Mềm

Bài 12: ***Các công cụ hỗ trợ kiểm tra phần mềm***

GV: Nguyễn Ngọc Tú

Email: nntu@hoasen.edu.vn

Bộ môn: Kỹ thuật Phần mềm

Nội dung

- Các dạng phần mềm hỗ trợ kiểm tra
- Sử dụng hiệu quả các công cụ
- Chọn lựa công cụ

Các dạng phần mềm

- Phân lớp
- Các công cụ quản lý kiểm tra
- Các công cụ quản lý yêu cầu
- Các công cụ quản lý lần vết
- Các công cụ quản lý cấu hình
- Các công cụ hỗ trợ rà soát
- Các công cụ hỗ trợ mô hình
- Các công cụ hỗ trợ thiết kế
- Các công cụ hỗ trợ chuẩn bị dữ liệu
- Các công cụ thực thi kiểm thử
- Các công cụ khác

Phân lớp các công cụ kiểm thử

- Có thể phân lớp theo hoạt động kiểm thử chính
- Phần lớn các phân lớp đều có hai dạng: thương mại và miễn phí
- Các công cụ tự động cho các công việc phải lặp đi lặp lại
  - Hiệu quả hơn
- Công cụ hỗ trợ giúp gia tăng sự tin cậy của phần mềm
  - So sánh kết quả, phát sinh tải, …
- Công cụ xâm nhập,..
- Một số công cụ giành riêng cho người phát triển

Các công cụ quản lý kiểm tra

- Cho phép lần vết các kiểm thử, kết quả kiểm thử, hỏng hóc, … tới cơ sở của kiểm tra
- Cho phép ghi nhận kết quả và xuất ra báo cáo
- Giúp quản lý kiểm tra và quá trình kiểm tra
- Cho phép giao tiếp tới các công cụ thực thi kiểm thử, quản lý, lần vết lỗi
- Quản lý phiên bản hoặc giao tiếp với công cụ quản lý cấu hình khác
- Phân tích định lượng (*metrics*) liên quan tới kiểm thử

Các công cụ quản lý yêu cầu

- Lưu trữ thông tin yêu cầu
- Kiểm tra sự đồng nhất và các yêu cầu lỗi
- Cho phép các yêu cầu được lập thứ tự ưu tiên
- Cho phép lần vết giữa các yêu cầu, chức năng, đặc tính với các mẫu thử.

Các công cụ lần vết

- Lưu trữ và quản lý các báo cáo về lỗi
- Phân lớp và lập ưu tiên cho lỗi thuận tiện
- Cung cấp luồng công việc trên cơ sở trạng thái bao gồm việc phân việc tới các thành viên
- Cho phép theo dõi lỗi và trạng thái thực hiện thông qua quá trình thực hiên
- Phân tích thống kê
- Tạo các báo cáo, báo biểu

Công cụ quản lý cấu hình

- Lưu trữ thông tin về các phiên bản, bản biên dịch (*build*) của phần mềm và *testware*
- Cho phép lần vết giữa *testware* và phần mềm cả những biến đổi của sản phẩm
- Hỗ trợ phát triển và kiểm thử trên nhiều môi trường HW/SW

Công cụ hỗ trợ Rà soát

- Lưu thông tin về quá trình rà soát
- Lưu và truyền thông các ghi chú rà soát
- Báo cáo khiếm khuyết và mức nỗ lực
- Quản lý các tham chiếu luật hoặc danh mục ý kiểm thử
- Hỗ trợ rà soát trực tuyến
- Lần vết giữa tài liệu và mã nguồn

Công cụ phân tích tĩnh

- Sử dụng chính bởi người phát triển
- Tìm khiếm khuyết trước khi kiểm thử động
- Tuân theo chuẩn mã hóa
- Phân tích cấu trúc và sự phụ thuộc
- Hỗ trợ để hiểu rõ mã nguồn
- Tính toán các yếu tố (*metrics*) từ mã

Các công cụ mô hình hóa và thiết kế

- Sử dụng chính bởi người phát triển
- Giúp tạo mô hình hệ thống
- Xác thực mô hình
- Hỗ trợ phát sinh một số TC từ mô hình

Các công cụ thiết kế kiểm thử

- Phát sinh thông tin đầu vào hoặc mẫu thử thực sự từ:
  - Yêu cầu
  - GUI
  - Mô hình thiết kế
  - Mã nguồn
- Phát sinh kết quả kỳ vọng
- Phát sinh frameworks, templates, and stubs cho kiểm thử

Các công cụ cho dữ liệu kiểm thử

- Thao tác hoặc tạo CSDL, tập tin, dữ liệu cho việc thực thi kiểm thử
- Tạo khối lượng lớn các dữ liệu hữu ích
- Chứng thực dữ liệu kiểm thử tùy thuộc vào các luật cụ thể
- Phân tích dữ liệu theo tần suất các điều kiện,…

Các công cụ thực thi kiểm thử

- Chạy mẫu thử
  - *gửi dữ liệu đầu vào qua đoạn script tự động và so sánh kết quả với kỳ vọng*
- Bao gồm bộ so sánh
- Tạo ra các ghi nhận có thể phân tích
- Thực hiện với GUI, API, CLI

Frameworks, Simulators cho kiểm thử

- Phần thay thế cho thiết bị phần cứng rắc rối có khả năng gây lỗi
- Kiểm thử đơn vị dễ dàng hơn bởi việc phát sinh và hỗ trợ bởi drivers, stubs, objects thay thế cho phần hoạt động của hệ thống
- Cung cấp các FW thực thi ở lớp giữa

Bộ so sánh kiểm thử

- Kiểm tra tập tin, csdl, kết quả kiểm thử so với kỳ vọng

Công cụ đánh giá bao phủ

- Sử dụng chính bởi người phát triển
- Đánh giá tỷ lệ % kiểu cấu trúc của mã đã được xét
  - Phát biểu
  - Nhánh / quyết định
  - Đối tượng
  - Gọi hàm
- Kiểm tra triệt để tới mức nào tập kiểm thử đã thực thi

Công cụ bảo mật

- Kiểm tra virus máy tính
- Mô phỏng các loại tấn công khác nhau
- Mô phỏng các điều kiện bảo mật khác nhau

Thực thi, giám sát và phân tích động

- Kiểm tra phụ thuộc thời gian và dò bộ nhớ
- Theo dõi và báo cáo hệ thống đã hoạt động ra sao dưới điều kiện mô phỏng
- Phát sinh các điều kiện tải khác nhau cho ứng dụng, csdl, mạng, máy chủ, …

Các công cụ khác

- Một số công cụ tập trung vào một ứng dụng cụ thể
  - Kiểm tra thực thi Web
  - Công cụ phân tích tĩnh cho từng ngôn ngữ
  - Công cụ kiểm tra bảo mật
- Miền ứng dụng đặc biệt
  - embedded systems
- Các công cụ gỡ rối

Hiệu quả sử dụng công cụ

- Những tác động
  - Khác biệt về kỹ thuật viết kịch bản kiểm thử
  - Lợi ích và rủi ro

Cơ hội và rủi ro

Cơ hội:

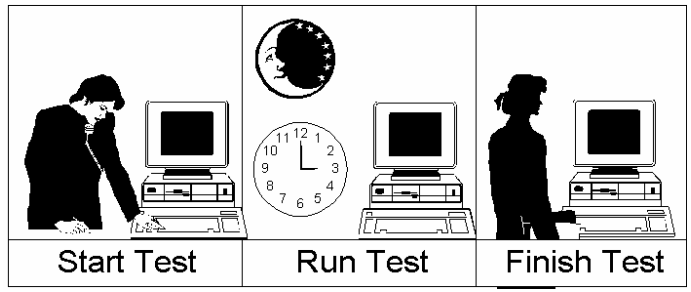
- Giảm công việc dư thừa
- Gia tăng sự đồng nhất và tính lặp
- Đánh giá mục tiêu
- Đơn giản hóa lập báo cáo

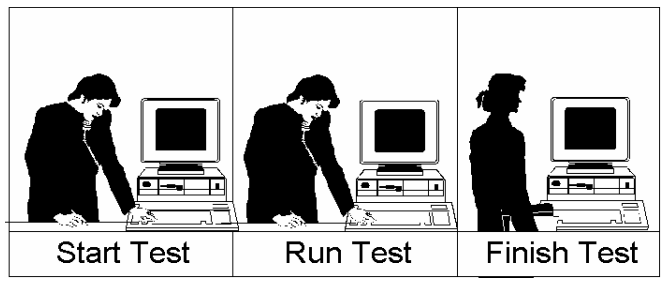
Rủi ro:

- Các kỳ vọng về phần mềm phi thực tế
- Không xác định được thời gian thực hiện, kiển thử, bảo trì, …
- Sử dụng công cụ cho các kiểm thử không phù hợp

Kiểm thử tự động

- Trong phát triển phần mềm , thông thường
  - Quy trình yêu cầu một lượng đáng kể về thời gian, kỹ năng và tiền bạc
- Theo thói quen, thường chỉ sử dụng phần mềm thực thi kiểm thử hơn là so sánh kết quả
  - Tự động hóa có thể phân tích biên dịch và gia tăng số lỗi tìm thấy





➔tự động hóa thông minh các công việc kiểm thử …

Kiểm thử thủ công và tự động

- Các kiểm thử phù hợp cho kiểm thử thủ công
  - Operations , maintenance
  - Configuration , compatibility
  - Error handling , recovery
  - Localization
  - Usability
  - Cài đặt
  - Tài liệu
- Kiểm thử bất kỳ hoặc kết hợp
  - Chức năng
  - Use cases (user scenarios)
  - Giao diện
  - Xử lý ngày tháng
- Các kiểm thử phù hợp cho kiểm thử tự động
  - Hồi quy, xác nhận
  - Ngẫu nhiên (Monkey , random)
  - Load, volume, capacity
  - Performance , reliability
  - Biên dịch chuẩn
  - white-box, API-based unit, component, integration
  - Static complexity and code analysis

Chú ý khi sử dụng công cụ

- Chiến lược không rõ ràng
- Kỳ vọng không thực tế
- Thiếu ký kết sử dụng, cung cấp để kiểm thử
- Chất lượng huấn luyện không tương xứng, nghèo nàn
- Đối tượng cho tự động hóa sai
- Chọn sai công cụ
- Tính khả dụng của công cụ
- Chọn sai nhà cung cấp
- Hệ thống dễ bị sụp đổ
- Thực hiện quá nhiều, quá sớm
- Không xác định về thời gian và tài nguyên cần thiết
- Môi trường kiểm thử đơn nhất hoặc không tương xứng
- Cách tiếp cận đúng nhưng sai thời điểm
- Chi phí quá mức

Rick Craig , Stefan Jaskiel's
Systematic Software Testing, Chapter 6

Đọc thêm

- [5]. Chapter 15

Q/A