

# SESSION 03

Các kỹ thuật kiểm tra

# Nội dung

Kiểm tra tĩnh và  
kiểm tra động

Kỹ thuật kiểm tra  
hộp trắng

Kỹ thuật kiểm tra  
hộp đen

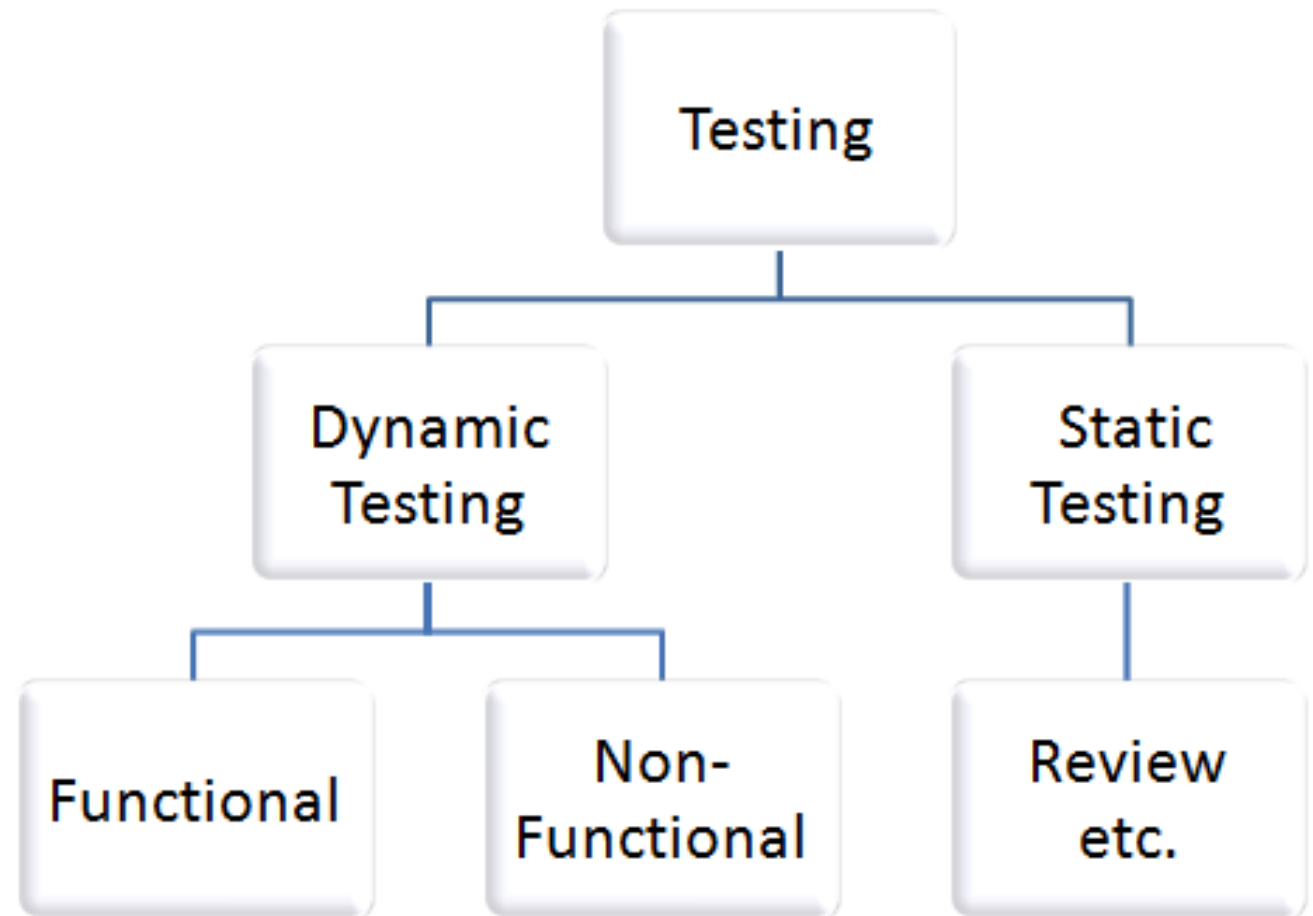
Tổng quan và sơ  
lược về CSDL SQL

Tổng quan và sơ  
lược về lập trình  
WEB












Tổng quan và sơ  
lược về lập trình  
Mobile

Tổng quan và sơ  
lược về lập trình  
API / WebServices

Kiểm tra tĩnh  
(Static Testing)  
và kiểm tra  
động (Dynamic  
Testing)



# Kiểm tra tĩnh (Static Testing) và kiểm tra động (Dynamic Testing)

 <b>Static Testing</b>	VS	 <b>Dynamic Testing</b>
ProfessionalQA.com™		
Involves the testing of the documented artefacts.	 INVOLVE	Involves the execution of the test cases over software product or code.
Analyzes and evaluates the software product in non-operational mode.	 WORKING	It performs the testing in the run time environment.
It is performed at an early stage of the development life cycle.	 WHEN	It is performed at a later stage of the development life cycle.
Verifies business requirement, functional requirement, etc.	 PURPOSE	Validates pre-specified requirements at each level of software development.
Works as a preventive measure to avoid defects in the software.	 WHY	Responsible for the corrective actions like removing & fixing defects.
Requires less amount of time and cost.	 TIME & COST	Time and cost, is comparatively higher.
Requires manual approach.	 APPROACH	May require manual or automation approach.
It is a Low Level activity.	 ACTIVITY	It is a High Level activity.
Statement Coverage is higher and almost near to 100%.	 STATEMENT COVERAGE	It does not guarantees the 100% statement coverage.

# Kỹ thuật tĩnh – Rà soát và quá trình kiểm tra

- Kỹ thuật tĩnh - Rà soát và quá trình kiểm tra
- Quy trình rà soát chương trình
  - Các dạng rà soát chương trình
  - Các giai đoạn rà soát
  - Các vai trò và trách nhiệm
  - Nhân tố thành công cho quá trình duyệt xét
- Phân tích tĩnh bằng các công cụ

# Khái niệm chính

---

- Khái niệm chính
  - ❑ Các sản phẩm thực hiện phần mềm và kỹ thuật kiểm tra
  - ❑ Tầm quan trọng và giá trị
  - ❑ Sự khác biệt giữa tĩnh và động

- Rà soát và công vụ
  - Rà soát theo các khía cạnh không chính thức và chính thức
  - Các công cụ có thể thực hiện vài kiểu kiểm tra tĩnh
  - Sử dụng cho
    - Yêu cầu, thiết kế, thêm mã, giản đồ csdl, tài liệu, các kịch bản, ...
- Mô hình và các khuôn mẫu
  - Sơ đồ của mô hình phức tạp có thể tiềm ẩn các vấn đề trong thiết kế
    - sơ đồ, cách dùng từ, ...
  - Một sơ đồ “xấu” đồng nghĩa với việc tiềm ẩn nhiều lỗi
- Kịch bản kiểm tra (*Test cases*) và dữ liệu
  - Kiểm tra phân tích và thiết kế trên cơ sở các đặc tả về PTTK
  - rà soát có cấu trúc
  - Kiểm tra phần này thường phát hiện nhiều “vấn đề”

# Các công cụ phân tích tĩnh

- Phân tích tĩnh
  - Từ khó hiểu: kiểm tra văn/ngữ pháp
  - Lập trình “nguy hiểm”: J-Test, Safer C, lint...
  - Đánh giá: phân tích độ phức tạp
- Mô phỏng hệ thống
  - Bộ giả lập hệ thống tổng quan
  - Các công cụ nghiên cứu mô hình/tác vụ thực thi
  - Bảng tính



# Chi phí & Lợi ích

- Chi phí
  - Thời gian
  - Sự nỗ lực thu thập và phân tích các yếu tố (*metrics*)
  - Cải tiến quá trình
- Lợi ích
  - Lịch biểu ngắn hơn (*do loại bỏ lỗi hiệu quả*)
  - Chu kỳ kiểm tra ngắn hơn, chi phí kiểm thử thấp hơn
  - Gia tăng năng suất
  - Cải tiến chất lượng sản phẩm
- Mấu chốt:
  - Những kỹ thuật có tính phản hồi cao nhằm cải tiến chất lượng












# Mối liên quan Tĩnh và Động

## Giống nhau

- Tìm- xác định khiếm khuyết
- Thực hiện tốt khi các phần liên quan phức tạp
- Tiết kiệm thời gian và tiền bạc

## Khác nhau

- Mỗi kỹ thuật tìm kiếm dạng khiếm khuyết khác nhau và hiệu quả hơn
- Kỹ thuật tĩnh nhằm tìm kiếm các thiếu sót hơn là tìm những hoạt động không mong đợi

 Static Testing	VS	 Dynamic Testing
Involves the testing of the documented artefacts.	 INVOLVE	Involves the execution of the test cases over software product or code.
Analyzes and evaluates the software product in non-operational mode.	 WORKING	It performs the testing in the run time environment.
It is performed at an early stage of the development life cycle.	 WHEN	It is performed at a later stage of the development life cycle.
Verifies business requirement, functional requirement, etc.	 PURPOSE	Validates pre-specified requirements at each level of software development.
Works as a preventive measure to avoid defects in the software.	 WHY	Responsible for the corrective actions like removing & fixing defects.
Requires less amount of time and cost.	 TIME & COST	Time and cost, is comparatively higher.
Requires manual approach.	 APPROACH	May require manual or automation approach.
It is a Low Level activity.	 ACTIVITY	It is a High Level activity.
Statement Coverage is higher and almost near to 100%.	 STATEMENT COVERAGE	It does not guarantees the 100% statement coverage.

# Các kiểu rà soát tĩnh

- Không chính thức:
  - Không có quy trình thực sự
    - Trao đổi ngoài lề, kiểm tra cho bạn, lập trình theo cặp
  - Chưa hữu ích, rẻ, phổ biến
- Ngang hàng (*đồng nghiệp*):
  - Quá trình loại bỏ khiếm khuyết đã xác định, lập tài liệu
  - Bao gồm các đồng nghiệp, chuyên gia kỹ thuật mà không có người quản lý

- “Lần bước” (*Walkthroughs*):
  - “lần’ theo tài liệu và mã
- “Thăm tra” (*Inspections*):
  - Người trung gian/điều phối (*đã được đào tạo*) đứng đầu một nhóm thăm tra thông qua một quá trình thăm tra chính thức
    - Danh mục ý kiểm định (*checklists*), mục thông tin (*entry*) , các tiêu chuẩn hiện có

- Ý không đầy đủ-mơ hồ che dấu ý nghĩa thực sự
- Cần đồng ý và hiểu nhất quán về đặc tả



# Quy trình rà soát tổng quát

- 
1. Lập kế hoạch
2. Khởi đầu (Kick-off)
3. Chuẩn bị
4. Gặp Xét duyệt
5. Làm lại/sửa lại
6. Tiếp tục
- Bao gồm: đánh giá, lập kế hoạch, tham gia huấn luyện, ...
- Lặp lại mỗi lần Rà soát một phần việc

Chi tiết phụ thuộc vào kiểu  
rà soát riêng cho từng dự án

Bao gồm các hoạt động cá nhân như toàn bộ việc phân tích cải tiến quy trình, đánh giá việc loại bỏ lỗi, ...

# Vai trò và trách nhiệm

- Điều phối (*Moderator*):
  - Chủ trì các cuộc họp
- Thư ký:
  - Tập hợp thông tin về tìm kiếm lỗi
- Tác giả:
  - Mô tả, giải thích và trả lời các câu hỏi
- Người rà soát (*Reviewer/inspector*):
  - Tìm kiếm lỗi
- Người quản lý:
  - Lập kế hoạch, sắp xếp tài nguyên và việc huấn luyện, hỗ trợ, phân tích các yếu tố quy trình
- Đôi khi, một người có thể đóng nhiều vai trò
  - Tác giả đôi khi đóng vai trò như Trung gian
  - Một trong những người rà soát làm thư ký
  - Đặc tả được xác định bởi kiểu rà soát

# Nhân tố rà soát thành công

- Cung cấp huấn luyện
- Rà soát cả sản phẩm – không sản phẩm
- Lập và theo sát lịch/mục tiêu
- Giới hạn tranh cãi
- Tập trung tìm kiếm “vấn đề”, không đi vào giải quyết
- Hiểu rõ các ghi chú đã viết
- Giới hạn, cẩn thận chọn lựa người tham gia
- Phải có sự chuẩn bị
- Lập danh sách “chú ý” kiểm tra
- Duyệt lại các phần Rà soát
- Đúng kỹ thuật
- Bảo đảm hỗ trợ quản lý
- Học – làm tốt hơn



# Lỗi thông thường trong yêu cầu-thiết kế

- **Mập mờ:** nghĩa chính xác là gì ?
  - VD: *Hệ thống nên cho phép đọc email ISP*
    - *ISP là gì ? Kích cỡ email ? Cho phép gửi kèm ?*
- **Không đầy đủ:** Rồi, Còn gì nữa ?
  - VD: *trên 3 lần nhập mật khẩu không đúng, hệ thống sẽ khóa tài khoản của NSD*
    - *Trong bao lâu ? Muốn mở khóa thì sao ? Ai có quyền mở khóa ?*
- **Không kiểm tra được:** Tôi có thể kiểm tra phần này ra sao ?
  - VD: *Hệ thống có khả năng sẵn sàng 100%*
    - *Không biết kỹ thuật kiểm tra để biết hoàn toàn sẵn sàng*
- **Phụ thuộc quá mức, kết nối và độ phức tạp**
  - Tìm kiếm các sơ đồ ‘tồi’ và gây khó hiểu

# Kiểm tra tĩnh

- Tìm khiếm khuyết trong mã và mô hình phần mềm
- Phân tích tĩnh thực hiện mà không cần thực thi hệ thống thực sự
- Phân tích tĩnh bao gồm việc phân tích hệ thống hay các thành phần bằng công cụ
- Phân tích tĩnh có thể tìm kiếm khiếm khuyết khó tìm hay khó cô lập
  - VD:
    - kết quả khả năng bảo trì, sử dụng con trỏ không an toàn
  - Dễ ràng cô lập lỗi vì tìm lỗi (*bugs*) chứ không thông qua “*triệu chứng*”

# Phân tích cái gì?

- Mã chương trình
  - Luồng điều khiển – luồng dữ liệu
- Mô hình chương trình
  - Mô phỏng
- Phát sinh kết quả dạng HTML & XML
- Tài liệu yêu cầu và thiết kế

## Lợi ích đem lại

- Phát hiện sớm, “rẻ”
- Cảnh báo về nhóm lỗi có thể tồn tại
  - nơi gây nguy hiểm, phức tạp cao, ...
- Định vị lỗi có thể bị bỏ qua khi kiểm tra động
- Phát hiện sự phụ thuộc thiếu nhất quán
- Nâng cao khả năng bảo trì mã/chương trình
- Ngăn ngừa khiếm khuyết trên cơ sở
  - Các yếu tố đã tập hợp
  - Các bài học từ việc phân tích

# Lỗi tiêu biểu

- Tham chiếu tới biến chưa gán trị
- Giao tiếp không nhất quán giữa chương trình và module
- Biến chưa bao giờ sử dụng
- Mã “chết”
- Vi phạm chuẩn lập trình
- Yếu điểm bảo mật
- Vi phạm cú pháp mã và mô hình
- V..V

# Công cụ sử dụng như thế nào?

- Người sử dụng tiêu biểu...
  - Người lập trình
    - Kiểm tra thành phần/tích hợp
  - Thiết kế và kiến trúc hệ thống
    - Thiết kế
- Đưa ra số lượng lớn các thông điệp cảnh báo
- Trình biên dịch có thể thực hiện một số phân tích tĩnh
  - Công cụ phân tích chuyên biệt hơn