Review sách chapter 16: Postdelivery Maintenance

I. Nội dung:

1. Tầm quan trọng của bảo trì sau khi giao hàng:

- Bảo trì sau giao hàng chiếm ít nhất 67% tổng chi phí vòng đời phần mềm, là giai đoạn tốn kém nhất.
- Bao gồm bảo trì khắc phục (corrective) để sửa lỗi, bảo trì hoàn thiện (perfective) để cải thiện chức năng hoặc hiệu suất, và bảo trì thích nghi (adaptive) để điều chỉnh theo thay đổi môi trường (ví dụ: hệ điều hành, phần cứng mới).
- Khác với phát triển, bảo trì khó hơn do phải làm việc với sản phẩm hiện có, thường thiếu tài liệu, và có thể không còn đội ngũ phát triển ban đầu.

2. So sánh phát triển và bảo trì:

- Phát triển tạo sản phẩm mới từ đầu, trong khi bảo trì sửa đổi sản phẩm hiện có, tương tự như chỉnh sửa một bức tranh đã hoàn thiện (ví dụ: thêm nhân vật vào bức chân dung).
- Bảo trì thường rẻ hơn (ví dụ: 5% chi phí phát triển lại), nhưng khó hơn do hạn chế về tài nguyên, tài liệu, và sự phức tạp của mã nguồn.

3. Thách thức của bảo trì:

- Thiếu tài liệu: Tài liệu thường không đầy đủ hoặc lỗi thời, buộc lập trình viên bảo trì phải dựa vào mã nguồn.
- Thiếu đội ngũ ban đầu: Lập trình viên bảo trì thường không tham gia phát triển ban đầu, dẫn đến khó khăn trong việc hiểu sản phẩm.
- Tâm lý coi thường: Bảo trì thường bị xem là công việc ít danh giá, giao cho lập trình viên mới hoặc kém kinh nghiệm, dù đòi hỏi kỹ năng cao.
- Lỗi hồi quy (regression faults): Thay đổi mã có thể gây ra lỗi mới ở phần khác của sản phẩm.
- Vấn đề mục tiêu di động (moving-target problem): Yêu cầu liên tục thay đổi khiến sản phẩm lệch khỏi thiết kế ban đầu, làm tăng độ phức tạp và chi phí bảo trì.

4. Kỹ năng cần thiết cho lập trình viên bảo trì :

- Lập trình viên bảo trì cần kỹ năng toàn diện: phân tích, thiết kế, lập trình, kiểm thử, và lập tài liệu.
- Kỹ năng gỡ lỗi (debugging) đặc biệt quan trọng để xác định lỗi trong sản phẩm lớn mà không có tài liệu đầy đủ.
- Cần khả năng làm việc độc lập và xử lý áp lực từ người dùng không hài lòng.

5. Quản lý bảo trì:

- Báo cáo lỗi (defect reports): Cần cơ chế ghi nhận và phân loại lỗi theo mức độ nghiêm trọng (critical, major, minor, v.v.). Lỗi nghiêm trọng cần sửa ngay, các lỗi khác có thể được xếp ưu tiên.
- Kiểm soát thay đổi: Mọi thay đổi cần được kiểm thử kỹ lưỡng, bao gồm kiểm thử hồi quy, và được phê duyệt bởi nhóm đảm bảo chất lượng (SQA) độc lập.

- Công cụ CASE: Các công cụ như Subversion, ClearCase, Bugzilla hỗ trợ kiểm soát phiên bản, theo dõi lỗi, và tái cấu trúc mã.

6. Bảo trì phần mềm hướng đối tượng:

- Hướng đối tượng (OOP) được cho là hỗ trợ bảo trì nhờ tính đóng gói (encapsulation) và ẩn thông tin (information hiding), giúp cô lập thay đổi.
- Tuy nhiên, các vấn đề như kế thừa (inheritance), đa hình (polymorphism), và ràng buộc động (dynamic binding) có thể làm phức tạp việc bảo trì, đặc biệt khi phải theo dõi các lớp trong hệ thống phân cấp lớn.

7. Kỹ thuật đảo ngược:

- Khi thiếu tài liệu, mã nguồn là tài liệu duy nhất. Kỹ thuật đảo ngược giúp tái tạo thiết kế hoặc yêu cầu từ mã nguồn, thường sử dụng công cụ như IBM Rational Rose hoặc Doxygen.
- Tái cấu trúc (restructuring) cải thiện mã mà không thay đổi chức năng, ví dụ: chuyển mã không có cấu trúc sang mã có cấu trúc.

8. Kiểm thử trong bảo trì:

- Kiểm thử quy hồi là bắt buộc để đảm bảo thay đổi không gây ra lỗi mới
- Cần lưu trữ tất cả các ca kiểm thử và kết quả mong đợi

9. Thách thức và giải pháp:

- Bảo trì khó hơn phát triển nhưng thường đượcđánh giá thấp. Cần thay đổi nhận thức bằng cách giao nhiệm vụ bảo trì cho các lập trình viên giỏi và trả lương xứng đáng
- Cần lập kế hoạch bảo trì từ giai đoạn phát triển, sử dụng tên biến rõ ràng, tài liệu đầy đủ và thiết kế linh hoạt.

II. <u>Bài học:</u>

- Bảo trì là giai đoạn quan trọng và tốn kém nhất
- Lập kế hoạch bảo trì từ giai đoạn phát triển
- Lập trình viên bảo trì cần kỹ năng toàn diện
- Kiểm thử hồi quy là bắt buộc
- Hướng đối tượng không tự động đảm bảo bảo trì dễ dàng
- Quản lý thay đổi yêu cầu chặt chẽ
- Vấn đề mục tiêu di động cần quản lý chủ động
- Công cụ CASE là cần thiết
- Thay đổi nhận thức về bao trì.
- Kỹ thuật đảo ngược là giải pháp cho hệ thống cũ.