Review sách chapter 17: More on UML

I. Nội dung:

1. UML là một ngôn ngữ, không phải phương pháp luận:

- UML là công cụ để diễn đạt ý tưởng, không giới hạn cách phát triển phần mềm hay phương pháp sử dụng (có thể áp dụng cho cả mô hình truyền thống và hướng đối tượng).
- UML đã trở thành tiêu chuẩn toàn cầu trong kỹ thuật phần mềm, với phiên bản mới nhất (tại thời điểm viết sách) là 2.0, được quản lý bởi Object Management Group (OMG)

2. Các biể đồ UML chính:

· **Class Diagrams (Biểu đồ lớp)**: Mô tả các lớp, thuộc tính, và quan hệ (kế thừa, tập hợp, liên kết). Bao gồm:

- **Aggregation (Tập hợp)**: Quan hệ phần-toàn (part-whole), ví dụ: xe hơi gồm khung, động cơ, bánh xe.
- **Multiplicity (Đa dạng)**: Số lượng đối tượng liên quan (ví dụ: 1, 0..1, 2..*, *).
- **Composition (Kết hợp)**: Dạng tập hợp mạnh hơn, nơi phần chỉ thuộc một toàn và bị xóa nếu toàn bị xóa.
- **Generalization (Tổng quát hóa)**: Quan hệ kế thừa.
- **Association (Liên kết)**: Quan hệ giữa các lớp, có thể là lớp liên kết (association class).

· **Notes (Ghi chú)**: Hình chữ nhật với góc gập để thêm bình luận, kết nối bằng đường nét đứt.
· **Use-Case Diagrams (Biểu đồ trường hợp sử dụng)**: Mô tả tương tác giữa diễn viên (actor) và hệ thống, hỗ trợ quan hệ **include** (bao gồm) và **extend** (mở rộng).
· **Stereotypes**: Cách mở rộng UML, sử dụng dấu « » (guillemets) để định nghĩa cấu trúc mới, ví dụ: «boundary», «include».
· **Interaction Diagrams (Biểu đồ tương tác)**: Bao gồm:

- **Sequence Diagrams (Biểu đồ tuần tự)**: Hiển thị tương tác giữa các đối tượng theo thời gian, với các yếu tố như tạo/hủy đối tượng, vòng lặp (*), tự gọi (self-call), và điều kiện (guard).
- **Communication Diagrams (Biểu đồ giao tiếp)**: Tương tự sequence diagrams nhưng tập trung vào liên kết giữa đối tượng.

· **Statecharts (Biểu đồ trạng thái)**: Mô hình trạng thái, sự kiện, và hành động. Hỗ trợ **guard** (điều kiện), **event** (sự kiện), **action/activity** (hành động/hoạt động), và **superstate** (siêu trạng thái) để giảm độ phức tạp.
· **Activity Diagrams (Biểu đồ hoạt động)**: Mô tả luồng công việc, đặc biệt khi các hoạt động diễn ra song song, sử dụng **fork** (phân nhánh), **join** (hợp nhất), và **swimlane** (vùng trách nhiệm).

· **Packages (Gói)**: Phân chia phần mềm thành các đơn vị độc lập, biểu diễn bằng hình chữ nhật với nhãn.
· **Component Diagrams (Biểu đồ thành phần)**: Hiển thị phụ thuộc giữa các thành phần phần mềm (mã nguồn, mã biên dịch, tệp thực thi).
· **Deployment Diagrams (Biểu đồ triển khai)**: Mô tả phần mềm được cài đặt trên phần cứng nào và các liên kết giao tiếp.

3. Tính linh hoạt của UML:

- UML hỗ trợ lặp và tăng dần (iteration and incrementation) trong Unified Process, cho phép bắt đầu với biểu đồ cơ bản và thêm chi tiết dần dần.
- Không yêu cầu sử dụng tất cả tính năng; người dùng có thể chọn tùy chọn phù hợp với dự án.

II. <u>Bài học:</u>

- UML là công cụ linh hoạt, hiểu và sử dụng UML là kỹ năng bắt buộc
- Tính linh hoạt của UML hỗ trợ phát triển lặp
- Lựa chọn biểu đồ phù hợp với mục đích