Chương 12

Hiện thực hướng đối tượng

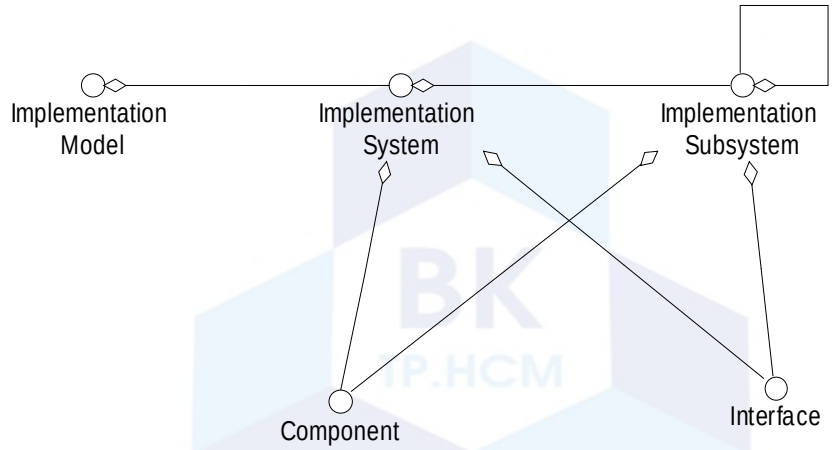
12.1 Nội dung công việc ở bước hiện thực

- Kế hoạch các bước tích hợp hệ thống phần mềm theo cơ chế tăng dần (hệ thống được thực hiện như chuỗi các bước nhỏ và dễ quản lý).
- Phân tán hệ thống phần mềm bằng cách ánh xạ các thành phần khả thi trên các nút trong mô hình triển khai (dựa chủ yếu vào các class chủ động).
- Hiện thực các class và hệ thống con thiết kế.
- Kiểm tra đơn vị trên các thành phần, tích hợp chúng vào 1hay nhiều file khả thi trước khi gởi đi kiểm tra tích hợp và kiểm tra hệ thống.

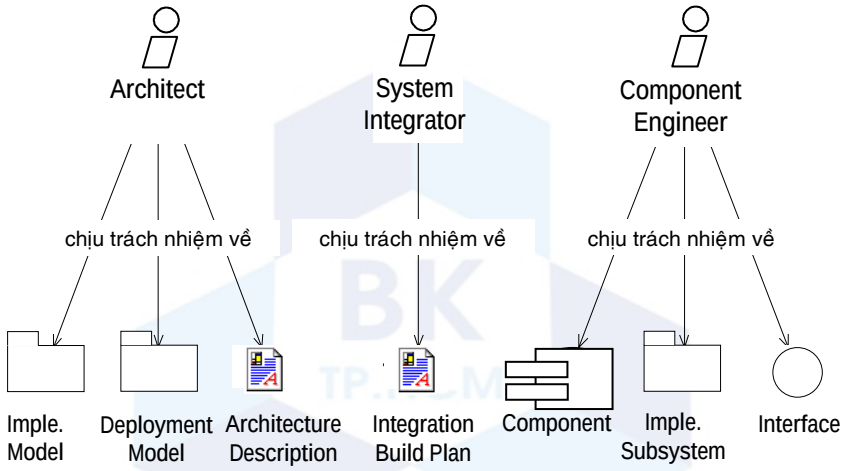
12.2 Các artifacts cần tạo ra trong hiện thực

- Mô hình hiện thực = hệ thống hiện thực :
  - Các hệ thống con hiện thực
  - Các component (executable, file, library, table, document,...)
  - Các interface của hệ thống con và component.
  - Kế hoạch tích hợp các "build"
  - Đặc tả kiến trúc (view of Implementation model)

12.2 Các artifacts cần tạo ra trong hiện thực



12.3 Các workers tham gia trong hiện thực



12.4 Stub

- Stub là 1 thành phần mà phần hiện thực chỉ ở mức độ "template" để phát triển hoặc kiểm thử thành phần khác phụ thuộc vào stub. Mức độ hiện thực của stub có thể là :
  - Null = {}
  - Chỉ trả về giá trị điển hình.
  - Chỉ hiện thực sơ lược các hoạt động thiết yếu.
  - Chỉ dùng thuật giải đơn giản nhất…
- Stub có thể tối thiểu số thành phần mới cần hiện thực cho mỗi version của hệ thống phần mềm, nhờ đó đơn giản hóa việc tích hợp và kiểm tra tích hợp.

12.5 Hệ thống con

- Hệ thống con được tạo ra do cơ chế đóng gói của môi trường hiện thực :
  - Thư mục các file trong C++
  - Package trong Java
  - Assembly trong C#...
- Hệ thống con có thể được dùng lại bởi nhiều phần mềm khác nhau.

12.6 Build

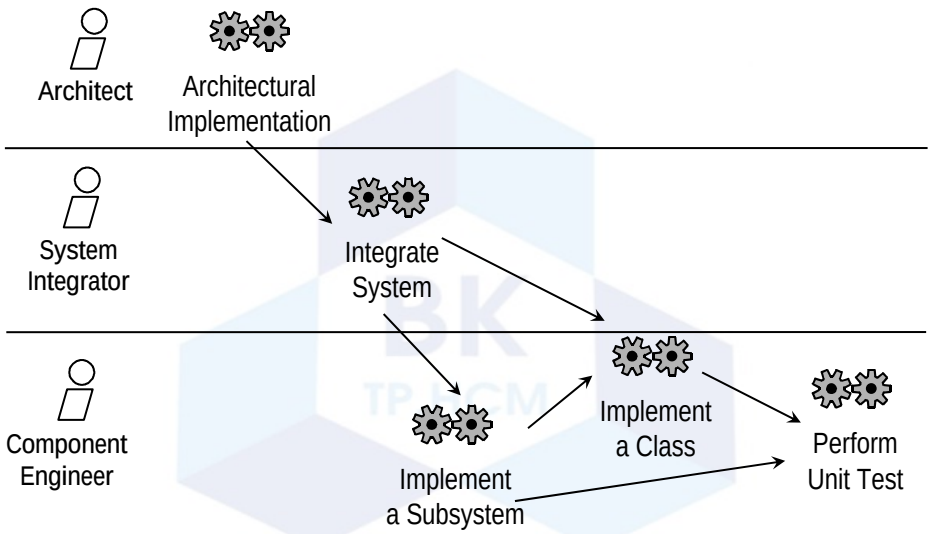
- Phần mềm được hiện thực theo cơ chế tăng dần theo từng bước nhỏ dễ quản lý, mỗi bước ta sẽ xây dựng 1 “Build”.
- Build là version khả thi của hệ thống phần mềm, cung cấp 1 số tính năng (thường chưa hoàn chỉnh) của phần mềm.
- Lợi ích của cách hiện thực theo các build là :
  - Có thể tạo ra version khả thi (thay vì phải đợi rất lâu), giúp kiểm thử tích hợp diễn ra sớm để demo cho các thành viên trong nhóm hay các người có liên quan xem.
  - Dễ dàng phát hiện lỗi hay điểm yếu vì mỗi lần chỉ có 1 phần mới khá nhỏ được thêm vào build cũ.
  - Kiểm tra tích hợp thường nhanh hơn là kiểm tra toàn bộ hệ thống phần mềm.

12.6 Build

Kế hoạch xây dựng các build miêu tả trình tự cụ thể việc xây dựng các build cho mỗi bước lặp, ứng với mỗi build, kế hoạch miêu tả :

- Chức năng kỳ vọng được hiện thực trong build là gì, đây là danh sách các use-case và/hoặc 1 số kích bản của chúng. Danh sách này cũng chỉ ra các yêu cầu phụ kèm theo
- Các phần nào của mô hình hiện thực bị tác động trong build, đây là danh sách các hệ thống con và các thành phần được đòi hỏi để hiện thực chức năng kỳ vọng của build.

12.7 Qui trình hiện thực



12.7 Qui trình hiện thực

Hiện thực kiến trúc :

Mục đích của hiện thực kiến trúc là phát họa mô hình hiện thực và kiến trúc của nó bằng cách :

- Nhận dạng các thành phần có ý nghĩa kiến trúc như các thành phần khả thi (exe).
- Ánh xạ các thành phần tới các nút trong cấu hình mạng liên quan : 1 class chủ động được hiện thực trong 1 thành phần khả thi và trở thành 1 process chạy ở 1 nút cụ thể…

12.7 Qui trình hiện thực

Tích hợp hệ thống :

Mục đích của tích hợp hệ thống là :

- Tạo 1 kế hoạch tích hợp các build, miêu tả build nào trong từng bước lặp và các yêu cầu trong mỗi build.
- Tích hợp mỗi build trước khi kiểm thử tích hợp nó.

Một vài tiêu chuẩn cho build kế tiếp là :

- Build nên thêm 1 số chức năng vào build trước bằng cách hiện thực hoàn chỉnh use-case đang hiện thực dang dở.
- Build không nên bao gồm quá nhiều thành phần mới hay được tinh chế.
- Dùng cơ chế nới rộng từ dưới lên để phát triển build kế tiếp.

12.7 Qui trình hiện thực

Tích hợp hệ thống :

Mục đích của tích hợp hệ thống là :

- Tạo 1 kế hoạch tích hợp các build, miêu tả build nào trong từng bước lặp và các yêu cầu trong mỗi build.
- Tích hợp mỗi build trước khi kiểm thử tích hợp nó.

Một vài tiêu chuẩn cho build kế tiếp là :

- Build nên thêm 1 số chức năng vào build trước bằng cách hiện thực hoàn chỉnh use-case đang hiện thực dang dở.
- Build không nên bao gồm quá nhiều thành phần mới hay được tinh chế.
- Dùng cơ chế nới rộng từ dưới lên để phát triển build kế tiếp.

12.7 Qui trình hiện thực

Hiện thực hệ thống con :

Mục đích của hiện thực hệ thống con là đảm bảo nó hoàn thành vai trò của mình trong mỗi build :

- Mỗi class trong hệ thống con thiết kế cần cho build hiện tại nên được hiện thực bởi thành phần trong hệ thống con hiện thực tương ứng.
- Mỗi interface của hệ thống con thiết kế cần cho build hiện tại cũng nên được cung cấp bởi hệ thống con hiện thực tương ứng.

12.7 Qui trình hiện thực

Hiện thực class :

Mục đích của hiện thực class là hiện thực từng class thiết kế thành file thành phần tương ứng, gồm các công việc sau :

- Phát họa 1 file thành phần chứa source code của class. 1 file có thể chứa nhiều class, nhưng để dễ quản lý và sử dụng, 1 file chỉ nên chứa 1 class.
- Tạo source code từ class thiết kế và các mối quan hệ mà class tham gia (dùng kỹ thuật round-trip của tiện ích CASE).
- Hiện thực các tác vụ của class thiết kế dưới dạng các method.
- Đảm bảo class hiện thực cung cấp đúng interface như class thiết kế.

12.7 Qui trình hiện thực

Khả năng tái sử dụng :

Khả năng tái sử dụng của từng tác vụ hay từng class phụ thuộc vào các yêu tố chính sau đây :

- Độ kết dính cao : nên thực hiện 1 chức năng rõ ràng, đủ nhỏ.
- Độ thống nhất cao : nên dùng cùng danh sách tham số cho các tác vụ có ý nghĩa sử dụng giống nhau.
- Tách biệt tác vụ thực thi và tác vụ chiến lược.
- Độ mở rộng cao : khái quát hóa kiểu tham số, số tham số.
- Tránh truy xuất dữ liệu toàn cục.

12.8 Kiểm thử đơn vị

Mục đích là kiểm thử chức năng của từng đơn vị (tác vụ) được hiện thực 1 cách riêng lẻ. Có 2 loại kiểm tra đơn vị :

- Kiểm thử hộp trắng (white-box testing) : kiểm thử chi tiết hiện thực bên trong đơn vị. Ta dùng kỹ thuật kiểm thử luồng điều khiển và kiểm thử đời sống của từng biến dữ liệu.
- Kiểm thử hộp đen (black-box testing) : kiểm thử hành vi đơn vị theo góc nhìn sử dụng từ ngoài, không biết gì về hiện thực bên trong. Ta dùng kỹ thuật kiểm thử dựa vào lớp tương đương, dựa vào các giá trị biên, dựa vào vùng miền, dựa vào bảng quyết định, dựa vào sự chuyển trạng thái, dựa vào use-case…

Kiểm thử đơn vị là chức năng của hoạt động hiện thực, workflow kiểm thử sẽ tiếp tục các mức độ kiểm thử khác như kiểm thử tích hợp, kiểm thử chức năng, kiểm thử hệ thống…

12.9 Kết chương

- Chương này đã giới thiệu các thông tin cơ bản về workflow hiện thực phần mềm.