Chương 7

Phân tích yêu cầu theo hướng đối tượng

7.1 Nhiệm vụ của phân tích yêu cầu chức năng

- Phát họa sơ lược cách thức giải quyết chức năng tương ứng. Nếu dùng kỹ thuật phân tích hướng đối tượng, bản phát họa cách giải quyết chức năng là các class đối tượng cụ thể, mối quan hệ giữa chúng và các thông tin kèm theo.
- Workflow phân tích yêu cầu sẽ xây dựng tất cả các bản phát họa cách thứ giải quyết mọi yêu cầu chức năng của hệ thống phần mềm.
- Toàn bộ các artifacts được tạo ra và duy trì trong workflow phân tích yêu cầu được gọi là mô hình phân tích.

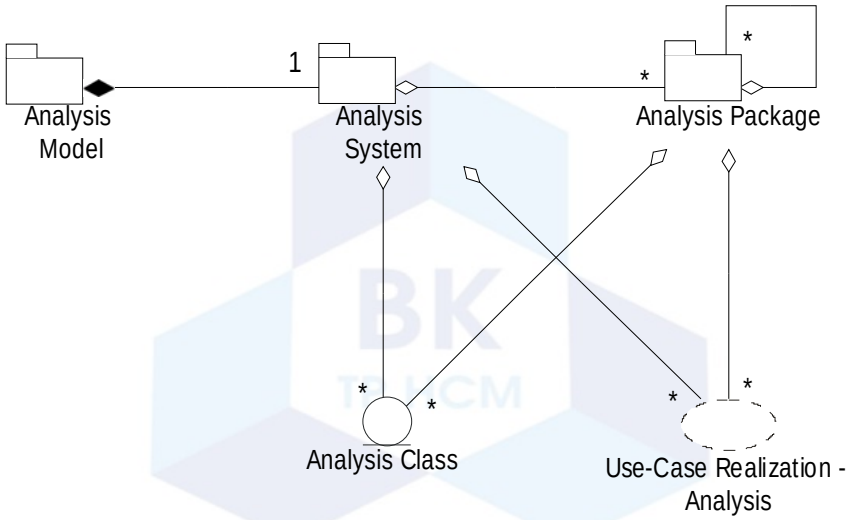
7.1 Nhiệm vụ của phân tích yêu cầu chức năng

- Mô hình phân tích có 1 số tính chất sau :
  - dùng ngôn ngữ của nhà phát triển để miêu tả mô hình sao cho dễ đọc, dễ hiểu, đơn nghĩa, rõ ràng…(ngôn ngữ UML).
  - Thể hiện góc nhìn từ bên trong hệ thống ở mức độ vĩ mô.
  - Được cấu trúc từ các class phân tích và, nếu cần, các package phân tích.
  - Được dùng chủ yếu bởi người phát triển để hiểu cách thức tạo hình dạng vĩ mô cho hệ thống phần mềm.
  - Cố gắng loại trừ mọi chi tiết dư thừa, không nhất quán.
  - phát họa cách hiện thực từng chức năng của hệ thống phần mềm.
  - Định nghĩa các dẫn xuất use-case, mỗi dẫn xuất use-case cấp phân tích miêu tả kết quả việc phân tích cho use-case đó.

7.2 Các artifacts cần tạo ra

- Mô hình phân tích = hệ thống các kết quả phân tích, nó chứa :
  - các package phân tích, nếu có, mỗi package chứa :
    - các dẫn xuất use-case ở cấp phân tích, mỗi dẫn xuất chứa :
      - các lược đồ class ở cấp phân tích.
      - các lược đồ tương tác giữa các đối tượng cấp phân tích.
      - 'flow of events' ở cấp phân tích
      - các yêu cầu đặc biệt của từng use-case, hay của toàn bộ các use-case
  - Đặc tả kiến trúc hệ thống phần mềm theo góc nhìn phân tích (view of analysis model)

7.2 Các artifacts cần tạo ra



7.2 Các artifacts cần tạo ra

- Mỗi lược đồ class ở cấp phân tích sẽ chứa nhiều class phân tích, nhưng chúng chỉ thuộc 1 trong 3 loại sau :
  - Class biên (boundary class) : mô hình sự tương tác giữa actor với hệ thống phần mềm. Nó miêu tả đối tượng giao tiếp giữa hệ thống phần mềm với thế giới bên ngoài, thí dụ như các đối tượng giao diện với người dùng phần mềm.
  - Class thực thể (entity class) : mô hình thông tin cần dùng. Nó miêu tả đối tượng chứa dữ liệu phục vụ cho chức năng tương ứng hoạt động. Đối tượng này có đời đống tương đối lâu dài và tầm vực sử dụng tương đối lớn trong hệ thống phần mềm.
  - Class điều khiển (control class) : mô hình việc xử lý, cộng tác giữa các đối tượng. Nó chứa các thuật giải xử lý hầu phục vụ chức năng tương ứng.

7.2 Các artifacts cần tạo ra

- Ký hiệu miêu tả các class phân tích :
  - Class biên (boundary class) :

  - Class thực thể (entity class) :

  - Class điều khiển (control class) :

7.2 Các artifacts cần tạo ra

- Mỗi lược đồ tương tác (trình tự, cộng tác) ở cấp phân tích sẽ chứa nhiều đối tượng ở cấp phân tích, nhưng chúng chỉ thuộc 1 trong 3 loại sau :
  - Đối tượng class biên (boundary class) :

name:classname

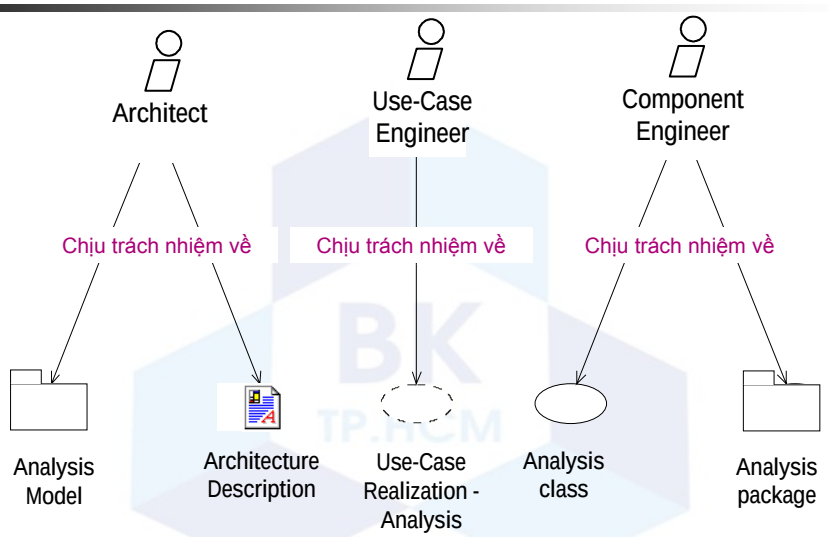
  - Đối tượng class thực thể (entity class) :

name:classname

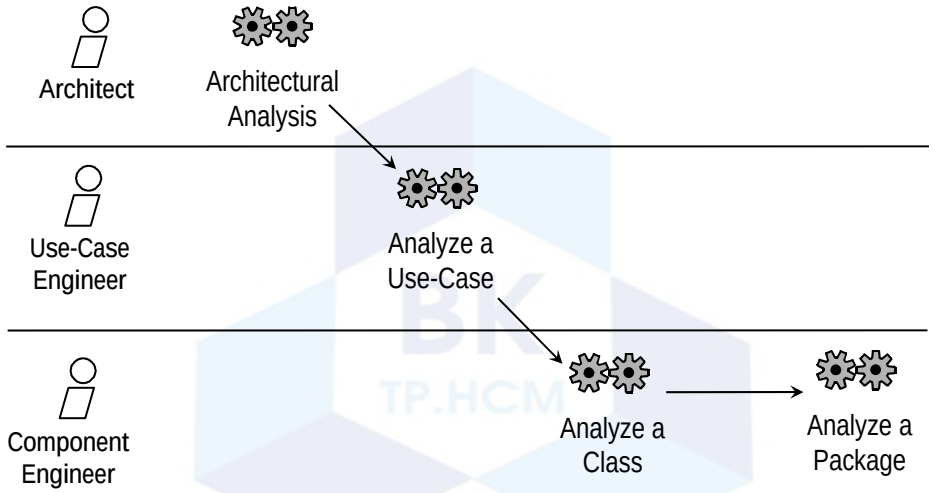
  - Đối tượng class điều khiển (control class) :

name:classname

7.3 Các worker tham gia phân tích yêu cầu



7.4 Qui trình phân tích yêu cầu



7.5 Phân tích kiến trúc

- Kiến trúc của 1 phần mềm nhỏ thì đơn giản, không có gì để nói (kiến triến căn chòi trẻ em chơi), tuy nhiên kiến trúc của 1 phần mềm lớn, phức tạp sẽ đóng vai trò rất quan trọng trong việc xây dựng và duy trì phần mềm theo thời gian (kiến trúc tòa nhà tháp hoa sen ở Q.1).
- Nhiệm vụ của hoạt động phân tích kiến trúc là phát họa mô hình phân tích và kiến trúc của hệ thống phần mềm thông qua các công việc cụ thể sau :
  - Nhận dạng các package phân tích
  - Nhận dạng các class phân tích dễ thấy
  - Nhận dạng các yêu cầu đặc biệt, các yêu cầu phi chức năng chung cho toàn bộ hệ thống phần mềm.

7.5 Phân tích kiến trúc

- Các package phân tích giúp ta tổ chức hệ thống thành những đơn vị nhỏ hơn theo cấu trúc cây phân cấp để dễ quản lý hế thống.
- Mỗi package chứa 1 số artifacts phân tích các use-case tương ứng. Các use-case trong từng package nên có những tính chất sau :
  - Chúng hỗ trợ cùng 1 qui trình nghiệp vụ xác định
  - Chúng hỗ trợ cùng 1 actor
  - Chúng có quan hệ mật thiết với nhau : tổng quát hóa, include, extend.
- Theo thời gian, khi việc phân tích tiến triển, ta sẽ tính chế cấu trúc của các package để ngày càng hợp lý hơn.

7.5 Phân tích kiến trúc

- Dựa vào thông tin về lĩnh vực và nghiệp vụ cần giải quyết trong workflow nắm bắt yêu cầu, ta dễ dàng nhận dạng và đề nghị 1 số class thực thể quan trọng nhất. Thí dụ trong phần mềm quản lý điểm SV, ta dễ dàng nhận dạng các class thực thể như class miêu tả SV, class miêu tả bảng điểm cho từng SV,…
- Các class phân tích còn lại sẽ được nhận dạng trong hoạt động phân tích từng use-case cụ thể.

7.5 Phân tích kiến trúc

- Các yêu cầu đặc biệt và phi chức năng quan trọng nhất cũng nên được nhận dạng để được lưu ý xử lý trong các bước sau. Chúng gồm :
  - Yêu cầu về mức độ bền vững
  - Yêu cầu về sự phân tán các thành phần và sự thi hành đồng thời giữa chúng.
  - Yêu cầu về an toàn dữ liệu.
  - Yêu cầu về mức đề kháng với lỗi
  - Yêu cầu về quản lý giao tác (transaction)
- Sau này, tính chất và mức độ của các yêu cầu đặc biệt và phi chức năng sẽ được cân nhắc lại trong từng class chức năng và từng dẫn xuất use-case.

7.6 Phân tích từng use-case

- Nhiệm vụ phân tích use-case là để :
  - Nhận dạng các class phân tích có đối tượng của mình tham gia vào việc thực hiện các hoạt động tồn tại trong "flow of events" của use-case tương ứng.
  - Thể hiện sự tương tác giữa các đối tượng phân tích trong việc thực hiện use-case thông qua các lược đồ động như lược đồ trình tự, lược đồ cộng tác, lược đồ hoạt động, lược đồ trạng thái.
  - Nhận dạng thêm 1 số yêu cầu đặc biệt và phi chức năng cho việc thực hiện use-case tương ứng. Cân nhắc lại tính chất và mức độ của các yêu cầu đặc biệt và phi chức năng chung đã nhận dạng được trong hoạt động phân tích kiến trúc.

7.6 Phân tích từng use-case

Nhận dạng các class phân tích thực hiện 1 use-case

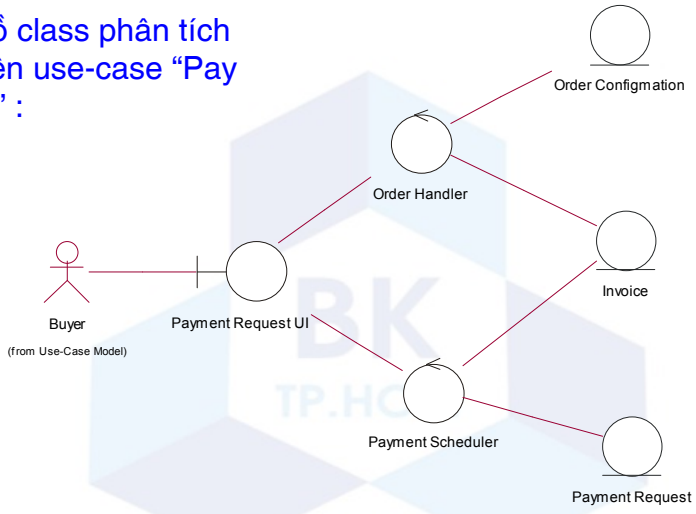
- Ở bước này, ta sẽ :
  - nhận dạng các class biên, thực thể, điều khiển cần thiết để thực hiện use-case tương ứng.
  - Phát họa tên, trách nhiệm, thuộc tính của từng class tìm được.
  - Nhận dạng các mối quan hệ giữa các class phân tích.
  - Tập hợp các class tìm được thành 1 hay nhiều lược đồ class. Các lược đồ class này sẽ là nội dung thiết yếu để xây dựng dẫn xuất use-case tương ứng.

7.6 Phân tích từng use-case

- Ta sẽ dùng các hướng dẫn sau đây để thực hiện phân tích từng use-case :
  - nhận dạng các class thực thể bằng cách chú ý các thông tin trong đặc tả use-case và trong mô hình lĩnh vực cần giải quyết của hệ thống phần mềm.
  - Nhận dạng class biên cho mỗi class thực thể vừa tìm được.
  - Ứng với mỗi actor là người dùng, hãy nhận dạng class biên trung tâm phục vụ cho sự tương tác người-chương trình.
  - Ứng với mỗi actor là hệ thống ngoài hay thiết bị I/O, hãy nhận dạng class biên trung tâm phục vụ cho sự tương tác chương trình-actor đó.
  - Nhận dạng class điều khiển có trách nhiệm xử lý các chức năng liên quan đến use-case tương ứng.

7.6 Phân tích từng use-case

Lược đồ class phân tích thực hiện use-case “Pay Invoice” :



7.6 Phân tích từng use-case

Xây dựng các lược đồ động

- Cần chú ý các điểm sau trong việc xây dựng các lược đồ tương tác giữa các đối tượng :
  - Đối tượng actor thường sẽ gởi thông báo đến class biên để kích hoạt use-case.
  - Mỗi class phân tích trong lược đồ class phải có ít nhất 1 đối tượng tham gia vào lược đồ động nào đó. Tương tự, mỗi đối tượng tham gia trong 1 lược đồ động phải thuộc 1 class phân tích nào đó trong lược đồ class phục vụ use-case.
  - Chưa vội kết hợp ngay 1 tác vụ cụ thể cho từng thông báo.
  - Các mối quan hệ giữa các đối tượng trong lược đồ động thường là "instance" của mối quan hệ kết hợp giữa các class tương ứng.

7.6 Phân tích từng use-case

Xây dựng các lược đồ động

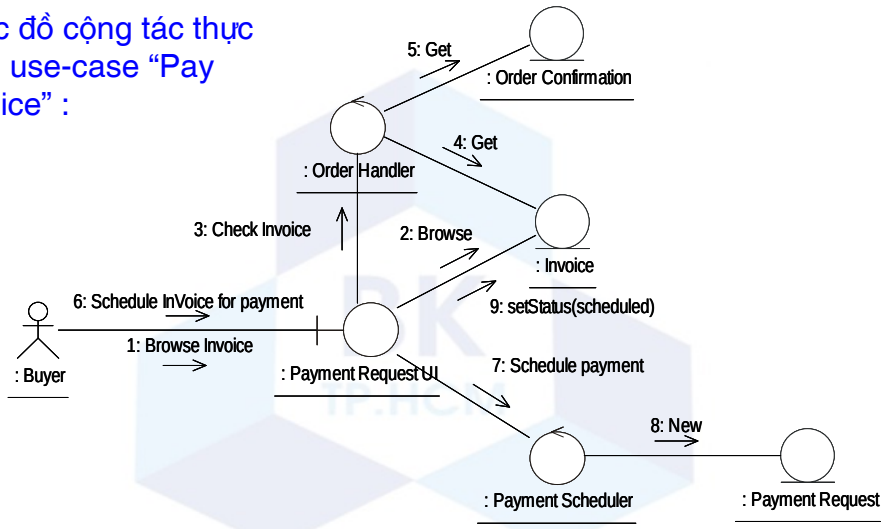
- Cần chú ý các điểm sau trong việc xây dựng các lược đồ tương tác giữa các đối tượng (tt) :
  - Chưa vội tập trung vào thứ tự thời gian xảy ra các thông báo giữa các đối tượng, nghĩa là lược đồ trình tự chưa cần thiết trong workflow phân tích.
  - Lược đồ cộng tác nên xử lý tất cả mối quan hệ giữa các đối tượng trong việc thực hiện use-case tương ứng.
  - Cần bổ sung đặc tả dạng văn bản cho lược đồ cộng tác, đặc tả này nên được để vào artifact "flow of events ở cấp phân tích".

7.6 Phân tích từng use-case

- Trong trường hợp cần xác định rõ thứ tự xảy ra các thông báo giữa các đối tượng, ta nên dùng các qui định sau :
  - Các thông báo được đánh số theo cấu trúc phân cấp :
    - 3.4.2 xảy ra sau 3.4.1 và cả 2 sẽ xảy ra trong lúc thực hiện thông báo 3.4.
    - 3.4.3a và 3.4.3b xảy ra đồng thời và được lồng trong thông báo 3.4
  - Dùng cú pháp tổng quát sau đây để miêu tả 1 thông báo :
    [precedessor] [guard-condition] [sequence-expression] [return-value :=] message-name argument-list
  - Ví dụ :
    2/ 1.3.1: p := find(specs)
    1.1, 4.2/ 3.2 *[i:=1..6]: invert(x, color)

7.6 Phân tích từng use-case

Lược đồ cộng tác thực hiện use-case "Pay Invoice" :



7.6 Phân tích từng use-case

Phân tích class

- Nhiệm vụ của việc phân tích từng class là :
  - Nhận dạng và duy trì các nghĩa vụ, trách nhiệm của class dựa vào vai trò của nó trong dẫn suất use-case.
  - Nhận dạng và duy trì các thuộc tính và các mối quan hệ của class với các phần tử khác.
  - Nhận dạng các yêu cầu đặc biệt và phi chức năng liên quan đến việc thực hiện class.

7.6 Phân tích từng use-case

Nhận dạng các nghĩa vụ, trách nhiệm của class

- Khi phân tích 1 class nào đó, lưu ý rằng nó có thể tham gia vào nhiều lược đồ class, lược đồ đối tượng thuộc nhiều dẫn suất use-case khác nhau. Do đó, ta phải nghiên cứu tất cả lược đồ class và lược đồ tương tác giữa các đối tượng trong các dẫn suất use-case mà có class tương ứng tham gia, từ đó tổng hợp tất cả nghĩa vụ, trách nhiệm của class và đối tượng thuộc class này trong các dẫn xuất use-case khác nhau.
- Đôi khi cần nghiên cứu "flow of events" ở cấp phân tích của nhiều dẫn xuất use-case khác nhau để tìm thêm nghĩa vụ và trách nhiệm của class tương ứng.

7.6 Phân tích từng use-case

Nhận dạng các thuộc tính của 1 class

- Mỗi nghĩa vụ, trách nhiệm thường cần 1 số thuộc tính, ta dùng các hướng dẫn sau để nhận dạng thuộc tính của 1 class :
  - Tên thuộc tính nên là danh từ.
  - Kiểu thuộc tính ở cấp phân tích nên ở mức ý niệm, chưa cần cụ thể hóa, nên dùng lại kiểu đã có khi đặc tả kiểu cho thuộc tính mới.
  - Nếu class phân tích quá phức tạp, nên tách 1 số thuộc tính phức tạp ra thành class riêng (class thực thể).
  - Thuộc tính của class thực thể thường dễ thấy hơn sơ với các class biên và điều khiển.

7.6 Phân tích từng use-case

Nhận dạng các thuộc tính của 1 class (tt)

- Mỗi nghĩa vụ, trách nhiệm thường cần 1 số thuộc tính, ta dùng các hướng dẫn sau để nhận dạng thuộc tính của 1 class :
  - Thuộc tính của class biên giao tiếp với người dùng thường miêu tả các thông tin được xử lý trực tiếp bởi người dùng, thi dụ field text, …
  - Thuộc tính của class biên giao tiếp với hệ thống ngoài thường miêu tả các tính chất của sự giao tiếp này, thí dụ ConnectionString miêu tả các thông tin để kết nối với database server.
  - Riêng các class điều khiển thì ta khó thấy thuộc tính của nó ở mức độ phân tích.

7.6 Phân tích từng use-case

Nhận dạng các tương tác giữa các đối tượng

- Các lược đồ tương tác sẽ miêu tả các tương tác giữa các đối tượng. Các tương tác này thường là 'instance' của mối quan hệ kết hợp giữa các class của chúng. Thí dụ trong lược đồ class, class A có 1 mối quan hệ kết hợp với class B thì trong lược đồ cộng tác (hay trình tự) nào đó, đối tượng class A sẽ phải tương tác với đối tượng class B.

7.6 Phân tích từng use-case

Nhận dạng các tương tác giữa các đối tượng

- Các mối quan hệ kết hợp có thể ám chỉ nhu cầu về sự gộp nhiều đối tượng thành một.
- Mối quan hệ bao gộp nên được dùng khi các đối tượng miêu tả :
  - Các khái niệm chứa vật lý khái niệm khác (xe chứa tài xế và khách).
  - Các khái niệm được xây dựng từ các khái niệm khác (xe gồm các bánh xe, động cơ,…).
  - Các khái niệm tạo thành tập hợp ý nhiệm nhiều đối tượng (gia đình gồm cha, mẹ, con…).
- Ta có thể rút trích các hành vi chung của nhiều class rồi định nghĩa class tổng quát hóa, đặt các hành vi chung vào class tổng quát hóa của chúng.

7.7 Phân tích các package

- Nhiệm vụ của việc phân tích các package là :
  - Đảm bảo từng package có tính độc lập với các package khác nhiều như có thể.
  - Đảm bảo từng package hoàn thành nhiệm vụ của mình là hiện thực đúng các class thuộc lĩnh vực liên quan hay các use-case liên quan.
  - Miêu tả các phụ thuộc sao cho có thể ước lượng ảnh hưởng của các thay đổi trogn tương lai.

7.7 Phân tích các package

- Dùng các hướng dẫn sau trong việc phân tích các package :
  - Đảm bảo từng package chứa đúng class phù hợp, cố gắng để tính kết dính cao (cohesion) bằng cách gộp các class có nhiều mối quan hệ chức năng lại với nhau.
  - Hạn chế tối đa sự phụ thuộc giữa các package, di dời các class quá phụ thuộc vào package khác đến package liên quan.

7.8 Kết chương

- Chương này đã giới thiệu các thông tin cơ bản về workflow phân tích yêu cầu chức năng như nhiệm vụ, các artifact cần tạo ra, các worker tham gia, qui trình thực hiện. Chương này còn giới thiệu chi tiết về hoạt động phân tích kiến trúc phần mềm và hoạt động phân tích từng use-case chức năng.