Phân tích thiết kế hệ thống với UML

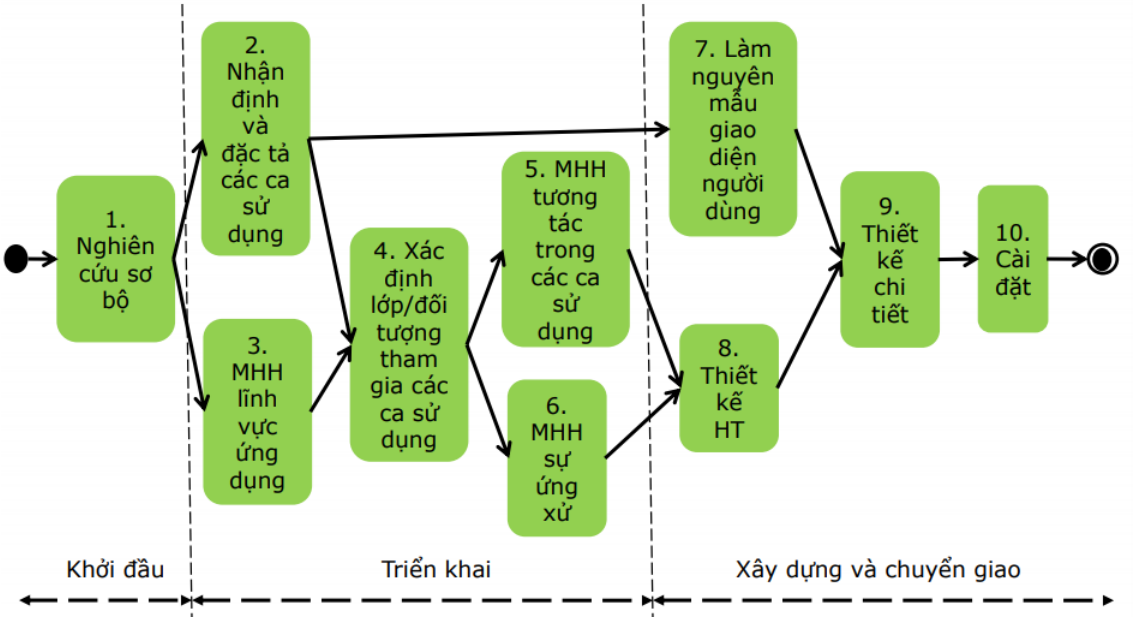
Ngô Quang Dương

Ngày 10 tháng 4 năm 2019

Tóm tắt nội dung

Trong tài liệu này, mình trình bày một cách ngắn gọn những điều cần biết về việc phân tích thiết kế hệ thống với **UML** - các loại sơ đồ **UML**, các giai đoạn trong phân tích thiết kế hệ thống và những việc được thực hiện trong những giai đoạn đó.

Mục lục



1 Các loại sơ đồ

1.1 Sơ đồ ca sử dụng

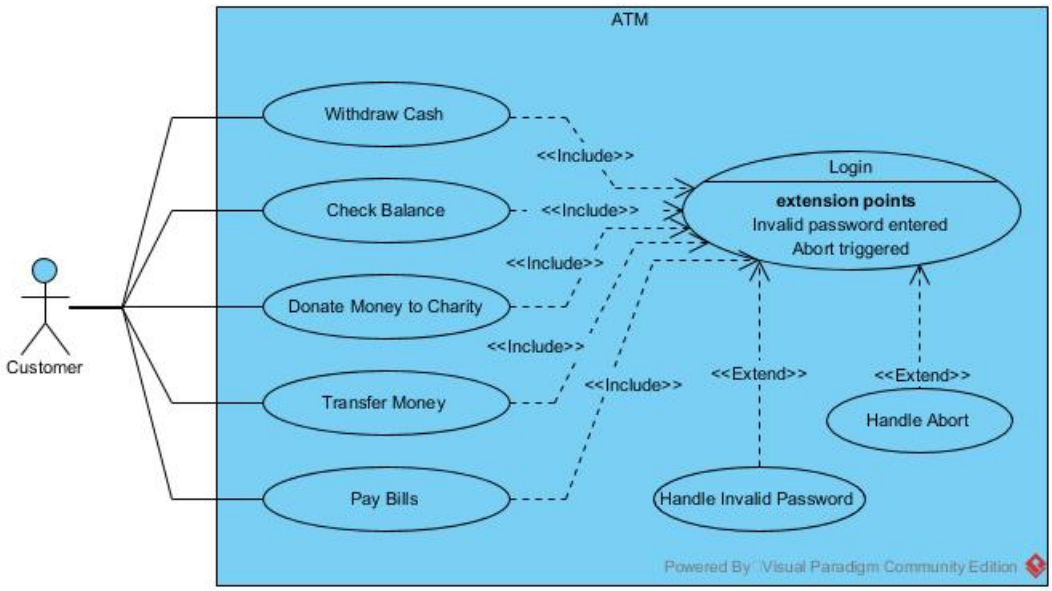
Thường được gọi nhiều hơn bằng cái tên *sơ đồ use case*, sơ đồ này thể hiện các *tác nhân* hệ thống (là những thành phần có tương tác với hệ thống nhưng không nằm trong hệ thống), các *chức năng* đối với từng tác nhân và quan hệ giữa các tác nhân với ca sử dụng.

Trong một *sơ đồ use case* thông thường, các *tác nhân* được biểu diễn bằng hình người que, còn các *chức năng*(use case) được biểu diễn bằng hình elip với nội dung là tên của *chức năng* đó.

Hãy xem qua ví dụ sau:

Giải thích các ký hiệu:

- *Khung hình chữ nhật*: đại diện cho hệ thống, cạnh của hình chữ nhật là biên hệ thống, những gì nằm trong là thành phần hệ thống, còn nằm ngoài thì không.
- *Người que*: tác nhân.
- *Hình elip*: use case.
- *Đường nét liền nối người que với hình elip*: thể hiện rằng tác nhân có thể sử dụng use case đó.
- *Mũi tên nét đứt với label «include»*: thể hiện rằng use case này(gốc mũi tên) chứa use case kia(ngọn mũi tên).
- *Hình elip có extension point*: use case có điểm mở rộng - tức là khi use case đó được thực hiện thì có thể có những tình huống khác xảy ra - như trong sơ đồ trên thì việc đăng nhập có thể đúng hoặc sai, mỗi tình huống đó là một extension point.
- *Mũi tên nét đứt với label «extend»*: thể hiện rằng use case này(gốc mũi tên) sẽ mở rộng use case kia(ngọn mũi tên), với điều kiện được trình bày trong extension point.



Hình 1: Sơ đồ use case cho hệ thống ATM

Sơ đồ use case có ý nghĩa quan trọng và xuyên suốt quá trình phá triển phần mềm:

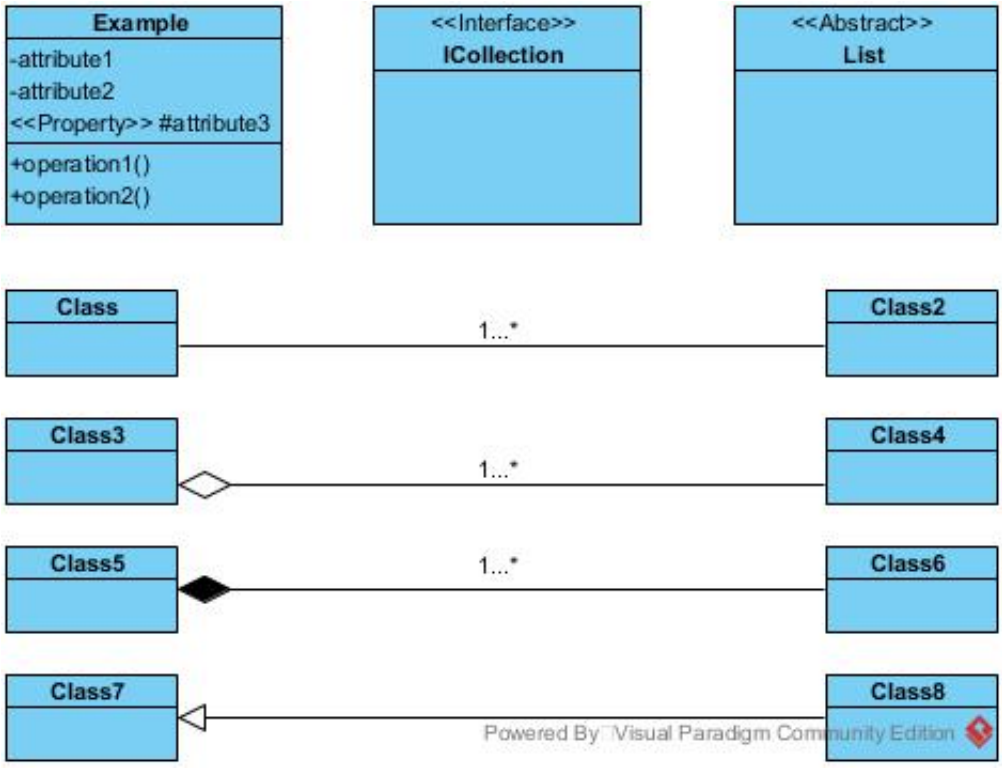
- Mang lại cái nhìn tổng quan về hệ thống, những việc cần làm đối với developer.
- Đối với người kiểm thử, sơ đồ use case giúp họ lập kế hoạch cho các tác vụ kiểm tra.
- Người viết tài liệu cũng dựa trên sơ đồ use case để viết hướng dẫn sử dụng.
- Người quản lý dự án sử dụng sơ đồ use case để lập kế hoạch

1.2 Sơ đồ lớp

Sơ đồ lớp (*class diagram*) là sơ đồ thể hiện nội dung của các lớp (thuộc tính, phương thức, bổ từ truy cập) và mối quan hệ giữa các lớp (kế thừa, kết tập, kết hợp, hợp thành).

Giải thích các ký hiệu:

- *Khung hình chữ nhật*:
  - *Khoang trên cùng*: tên class, phần «...» trên cùng là stereotype – để thể hiện những kiểu mà UML không có sẵn như là interface, abstract, . . . nếu không có gì thì mặc định hiểu đó là class.
  - *Khoang thứ hai*: danh sách các thuộc tính (có thể bổ sung kiểu dữ liệu).
  - *Khoang thứ ba*: danh sách các phương thức (có thể bổ sung kiểu dữ liệu trả về và các tham số cùng kiểu dữ liệu tương ứng).
  - *Dấu -*: `private`.
  - *Dấu #*: `protected`.
  - *Dấu +*: `public`.
- *Đường nét liền*: thể hiện quan hệ kết hợp - tức là hai class có tương tác với nhau.



Hình 2: Minh họa các ký hiệu trong sơ đồ lớp

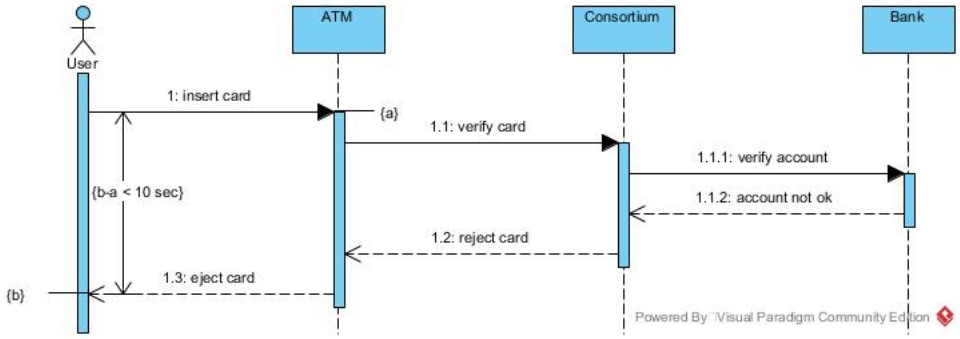
- *Đường nét liền có một đầu hình thoi trắng*: thể hiện quan hệ kết tập – class này(có hình thoi trắng) sở hữu một object của class kia(không có hình thoi trắng) nhưng object của class kia có thể tồn tại độc lập với class này.
- *Đường nét liền có một đầu hình thoi đen*: thể hiện quan hệ hợp thành – cũng tương tự như quan hệ kết tập nhưng khác là object của class kia chỉ tồn tại nếu có object của class này.
- *Bội số(1...*, *, 1*: trong mối quan hệ kết hợp với bộ 1 thì object của class bên trái sẽ kết hợp với 1 object của class bên phải; 1...* thì object của class bên trái kết hợp với ít nhất 1 object của class bên phải; * thì object bên trái có thể kết hợp với không hoặc nhiều object của class bên phải.
- *Mũi tên có ngọn hình tam giác trắng*: thể hiện quan hệ tổng quát hóa – class ở ngọn mũi tên tổng quát class ở gốc mũi tên.

1.3 Sơ đồ tuần tự

Sơ đồ tuần tự (sequence diagram) được sử dụng để mô tả các thao tác được thực hiện trong một use case – cụ thể là các object tương tác với nhau như thế nào, gọi tới những phương thức nào.

Các bước của quá trình tương tác rất tiên theo dõi vì chúng được đặt theo thứ tự từ trái sang phải, từ trên xuống dưới và còn được đánh số, sử dụng các mũi tên.

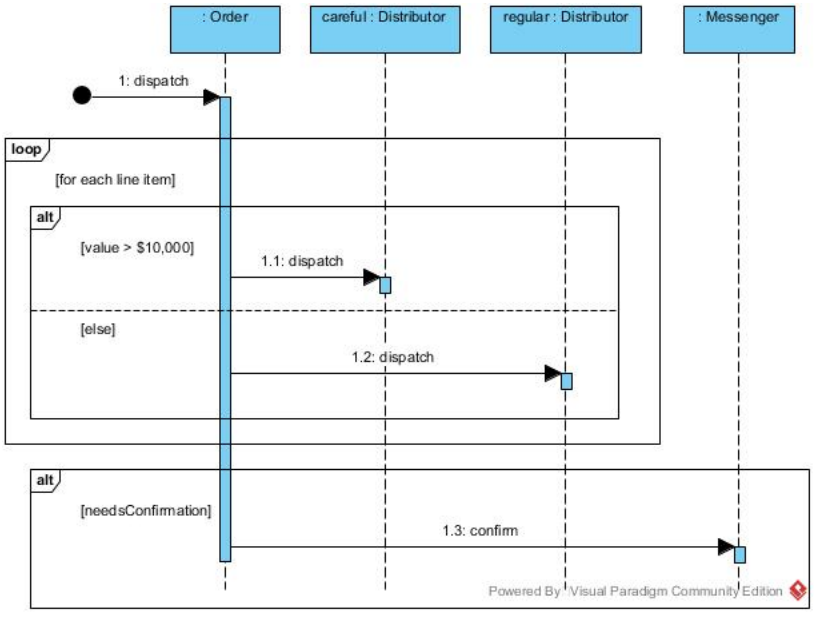
Hãy quan sát sơ đồ tuần tự thể hiện quá trình cho thẻ vào máy ATM :



Hình 3: Sơ đồ tuần tự minh họa việc cho thẻ vào máy ATM

Giải thích các ký hiệu:

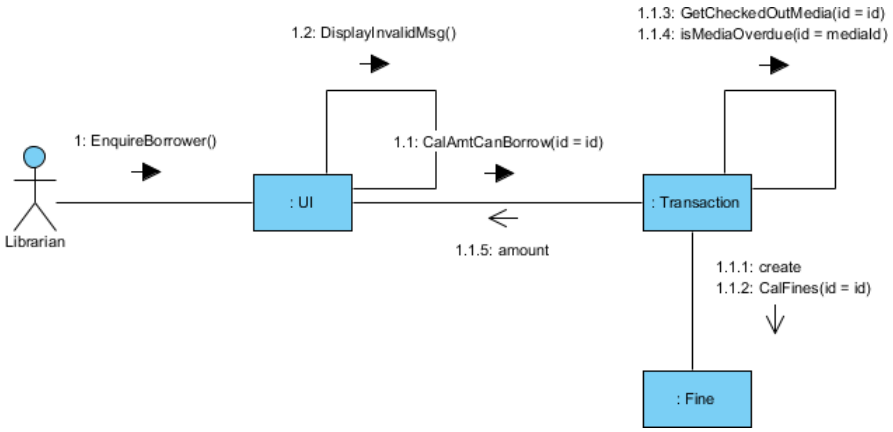
- *Hình người que*: tác nhân.
- *Hình chữ nhật có nhãn*: object.
- *Đường nét đứt thẳng*: đường sống (life line).
- *Hình chữ nhật dài trên đường sống*: thời gian hoạt động của object hoặc tác nhân đó.
- *Mũi tên nét liền*: gọi phương thức của object được trỏ đến.
- *Mũi tên nét đứt*: kết quả trả về.



Hình 4: Ví dụ về cấu trúc lặp và rẽ nhánh trong sơ đồ tuần tự

Có thể thấy rằng thứ tự được gọi của các phương thức được thể hiện qua cách đánh số và cách sắp xếp từ trái sang phải, từ trên xuống dưới. Ngoài ra, trong sơ đồ tuần tự, người ta còn có thể trình bày các cấu trúc lặp, rẽ nhánh, `continue`, `break`. Trong đó, cấu trúc lặp, rẽ nhánh sẽ được đặt vào một khung có nhãn lần lượt là **loop** và **alt**.

1.4 Sơ đồ giao tiếp



Hình 5: Sơ đồ giao tiếp thể hiện quá trình mượn sách

Sơ đồ giao tiếp(communication diagram) có tác dụng giống hệt như sơ đồ tuần tự. Ưu điểm của nó so với sơ đồ tuần tự là sơ đồ giao tiếp có thể trình bày tự do hơn, trong khi sơ đồ tuần tự có một cấu trúc cố định (sẽ càng chiếm nhiều diện tích hơn khi số lượng object hoặc các bước tăng lên). Nhược điểm là nó không tiện theo dõi thứ tự các bước như trong sơ đồ tuần tự.

1.5 Sơ đồ hoạt động

Sơ đồ hoạt động(activity diagram) khá giống với sơ đồ luồng, sơ đồ khối (flowchart). Nếu như sơ đồ tuần tự và sơ đồ giao tiếp thể hiện các object và lời gọi các phương thức thì ở sơ đồ hoạt động, quá trình đó được thể hiện chi tiết hơn – có thể hiểu là sơ đồ này sẽ trình bày cả những điều được thực hiện bên trong phương thức.

Sơ đồ hoạt động luôn bắt đầu bằng một node bắt đầu (một hình tròn đen) và đích đến là node kết thúc (hình tròn đen có viền). Một sơ đồ hoạt động luôn có đúng một node bắt đầu nhưng có thể có nhiều node kết thúc (tương ứng với nhiều tình huống khác nhau).

Các bước của một quá trình được thể hiện trong sơ đồ hoạt động bằng các hình chữ nhật có bo tròn góc. Các mũi tên sẽ trỏ luồng hoạt động. Cũng giống như trong sơ đồ khối, sơ đồ hoạt động cũng có thể thể hiện cấu trúc rẽ nhánh (bằng node quyết định hình thoi) và cấu trúc lặp.

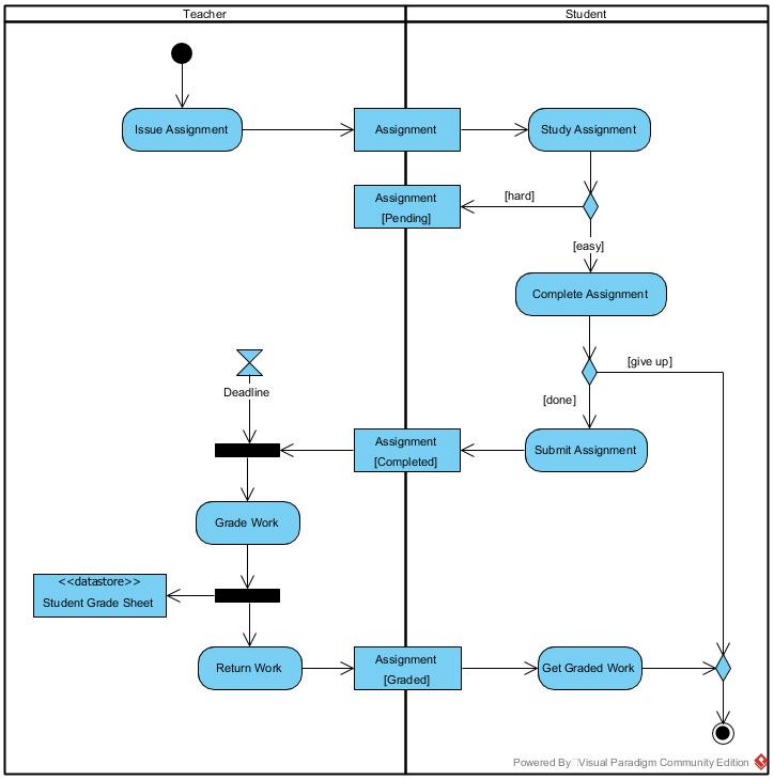
1.6 Sơ đồ máy trạng thái

Trong lập trình hướng đối tượng, một đối tượng có hai đặc tính: thứ nhất là thuộc tính(property or attribute), thứ hai là phương thức(method or operation). Trong đó, thuộc tính cũng chính là *trạng thái* của đối tượng.

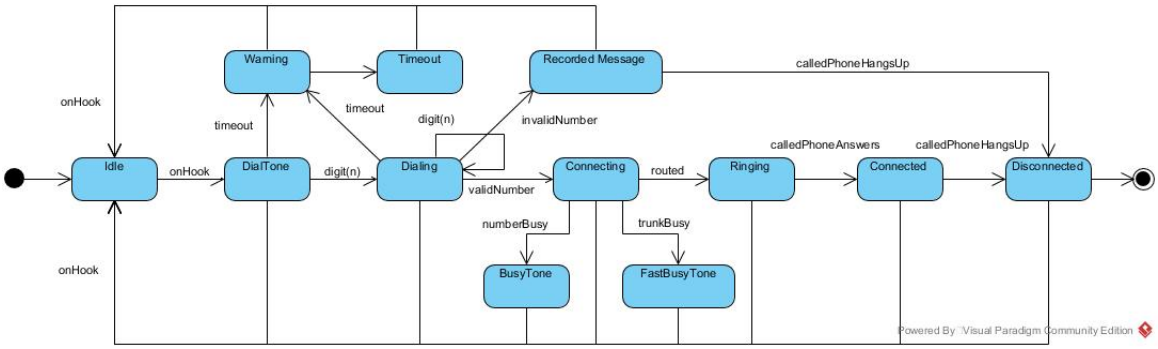
Sơ đồ máy trạng thái(state machine diagram) chỉ ra trạng thái của hệ thống và những sự kiện, phương thức có thể kích hoạt sự thay đổi của những thuộc tính đó.

1.7 Sơ đồ quan hệ thực thể

Sơ đồ quan hệ thực thể(entity relationship diagram) thể hiện mối quan hệ giữa các thực thể được lưu trữ trong một cơ sở dữ liệu. Thực thể trong ngữ cảnh này chính là những object. Một tập thực thể là một bộ các thực thể cùng loại. Các thực thể có thể có một hoặc nhiều thuộc tính.



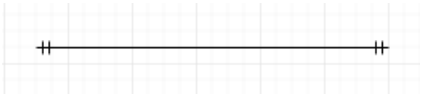
Hình 6: Sơ đồ hoạt động



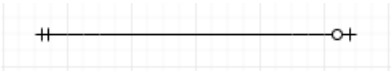
Hình 7: Minh họa sơ đồ máy trạng thái cho hệ thống điện thoại

Mối quan hệ trong sơ đồ quan hệ thực thể có thể thuộc một trong các loại sau, còn được gọi là những bội số. Bội số nếu có thể nhận giá trị 0 được gọi là *optional*(tùy chọn), ngược lại được gọi là *mandatory*(bắt buộc).

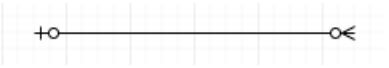
- **1 – 1**. Quan hệ một – một, được thể hiện bằng đường nối:
- **1 – n**. Quan hệ một – nhiều:
- **n – n**. Quan hệ nhiều – nhiều:



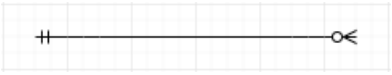
Hình 8: 1 – 1



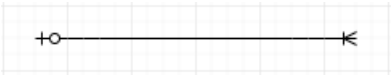
Hình 9: 1 – 0. . . 1



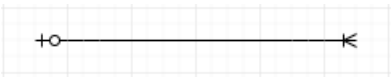
Hình 10: 0..1 – 0..n



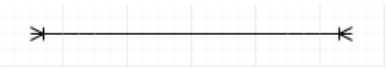
Hình 11: 1 – 0..n



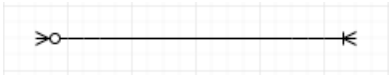
Hình 12: 0..1 – n



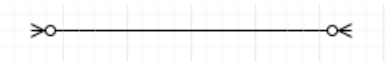
Hình 13: 1 – n



Hình 14: n – n



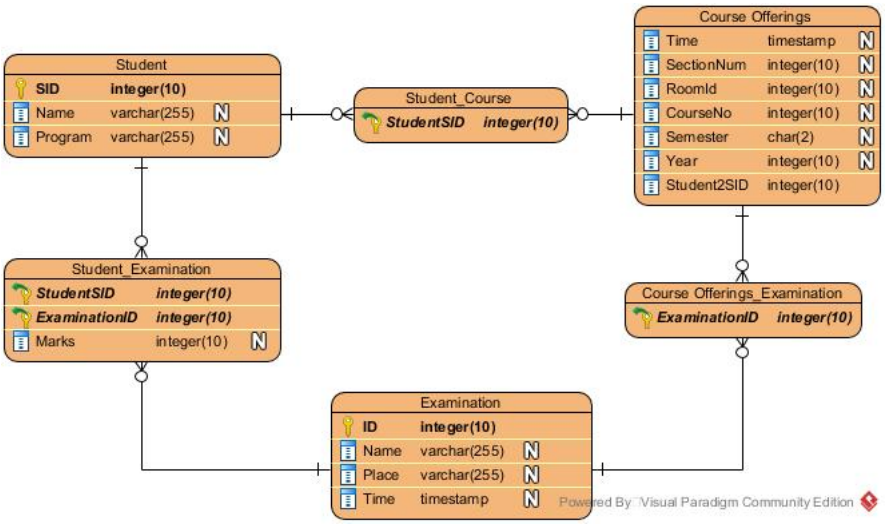
Hình 15: 0..n – n



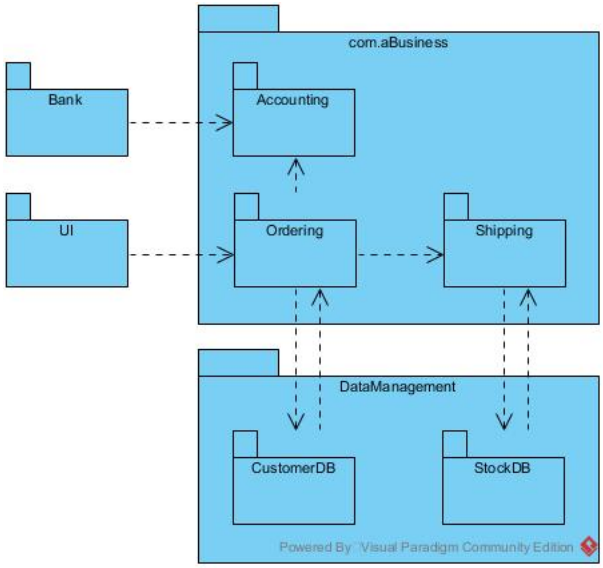
Hình 16: 0..n – 0..n

1.8 Sơ đồ gói

Một tính chất khác của lập trình hướng đối tượng là tính đóng gói. Có hai khía cạnh của tính đóng gói: thứ nhất là sự giới hạn truy cập một thuộc tính, phương thức của đối tượng; thứ hai là sự nhóm họp của các class có cùng chức năng. Khía cạnh thứ nhất đã được thể hiện trong *sơ đồ lớp*, còn khía cạnh thứ hai được thể hiện trong sơ đồ gói(package diagram)



Hình 17: Ví dụ sơ đồ quan hệ thực thể cho cơ sở dữ liệu lưu trữ điểm của sinh viên



Hình 18: Sơ đồ gói

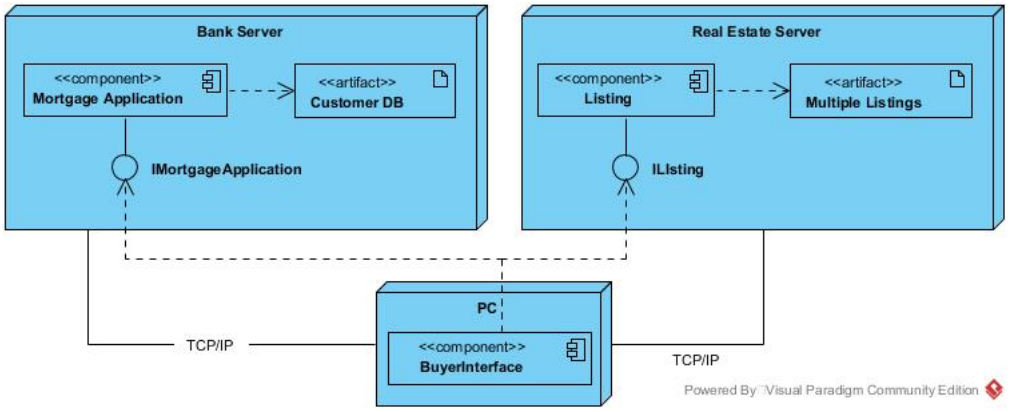
Giải thích các ký hiệu:

- *Hình tập tài liệu*: đại diện cho một package, các package có thể rời nhau hoặc package này chứa một hay nhiều package con.
- *Mũi tên nét đứt*: quan hệ phụ thuộc, package ở gốc mũi tên phụ thuộc vào package ở ngọn mũi tên.

Ngoài ra trong các package trên sơ đồ gói có thể biểu diễn thêm các class, interface bên trong.

1.9 Sơ đồ triển khai

Sơ đồ triển khai (deployment diagram) được sử dụng để mô tả sự bố trí các *thành phần thực thi*.



Hình 19: Sơ đồ triển khai

Các thành phần đó có thể là phần cứng(nút cứng, các thiết bị vật lý, ...) hoặc phần mềm(nút mềm, các file thực thi, server, ...). Các nút trong một sơ đồ triển khai được kết nối với nhau.

Giải thích các ký hiệu:

- *Hình hộp*: một nút.
- *Các stereotype*: các thành phần trong một nút.
- *Hình tròn*: giao diện(interface).
- *Mũi tên nét đứt*: thể hiện việc sử dụng, cài đặt giao diện.

2 Các giai đoạn trong phân tích thiết kế

2.1 Thu thập và phân tích yêu cầu

2.1.1 Định nghĩa yêu cầu

Yêu cầu hệ thống, hay ngắn gọn là **yêu cầu** là những gì mà hệ thống phải có, phải thực hiện. Những yêu cầu gồm:

- Những chức năng mà hệ thống phải có.
- Những đặc điểm mà hệ thống phải có.
- Thông tin về nghiệp vụ.

Những yêu cầu có thể thay đổi trong quá trình phát triển.

Có hai loại yêu cầu: *yêu cầu chức năng* và *yêu cầu phi chức năng*. Yêu cầu chức năng liên quan trực tiếp tới tiến trình mà hệ thống thực hiện. Còn yêu cầu phi chức năng là những tính chất hành vi mà hệ thống phải có như là: khả năng, năng suất, tính bảo mật, yêu cầu pháp lý, ...

2.1.2 Tiêu chuẩn đánh giá yêu cầu

Một yêu cầu tốt sẽ đáp ứng những tiêu chuẩn sau:

- *Nhất quán* – các yêu cầu không mâu thuẫn.
- *Đầy đủ.*
- *Khả thi.*
- *Cần thiết* – đáp ứng mục đích của hệ thống.
- *Chính xác* – yêu cầu phải được phát biểu chính xác.
- *Dễ theo dõi* – các yêu cầu đều tương ứng với chức năng và đặc trưng nào đó của hệ thống.
- *Dễ kiểm tra* – các yêu cầu được xác định có thể kiểm tra được trong giai đoạn kiểm thử.

2.1.3 Các phương pháp thu thập yêu cầu

- *Nghiên cứu tài liệu*: sổ sách, hồ sơ, báo cáo, thống kê, các file trên máy, ...
- *Quan sát.*
- *Phỏng vấn*: lựa chọn người thích hợp để phỏng vấn, đưa ra các câu hỏi (câu hỏi đóng, câu hỏi mở hoặc câu hỏi thăm dò), tiến hành, báo cáo kết quả phỏng vấn.
- *Dùng phiếu hỏi.*

Việc thu thập yêu cầu cần phải trung thực, khách quan, phản ánh đúng thực tế, không bỏ sót thông tin, không trùng lặp.

2.2 Đặc tả yêu cầu

2.2.1 Bảng thuật ngữ(glossary)

Trong một nghiệp vụ, một dự án, người ta thường sử dụng những thuật ngữ riêng biệt. Thuật ngữ có thể là một từ, những chữ viết tắt. Việc hiểu và sử dụng những thuật ngữ này giúp việc trình bày, nắm bắt vấn đề được ngắn gọn, nhanh chóng.

2.2.2 Xây dựng sơ đồ use case

Sau khi thu thập và đánh giá những yêu cầu đó, chúng ta sẽ thực hiện việc mô hình hóa chúng. Những yêu cầu sẽ không được trình bày bằng lời mà bằng *sơ đồ use case.*

Các use case biểu diễn những chức năng của hệ thống nhưng không cần chỉ rõ chúng được thực hiện như thế nào.

Tham khảo phần sơ đồ use case được trình bày trong chương 1. Các bước để lập sơ đồ use case:

- *Xác định tác nhân*: Tác nhân có thể là người hoặc vật thể không thuộc về hệ thống nhưng có tương tác với hệ thống. Tác nhân có thể là khách, người quản lý, người bảo dưỡng, các thiết bị ngoài hoặc những hệ thống khác.
- *Xác định use case*: Use case là một chuỗi hành động của hệ thống nhằm thu được kết quả cho một tác nhân nào đó. Use case xác định một chức năng dành cho tác nhân. Use case phải liên kết tới ít nhất một tác nhân.
- *Xác định mối quan hệ giữa các tác nhân với nhau, các use case với nhau và giữa tác nhân với use case.*

2.2.3 Bản mô tả use case

Sơ đồ use case thể hiện các tác nhân và chức năng một cách xúc tích, ngắn gọn. Tuy nhiên ta vẫn cần đến những bản mô tả/đặc tả use case. Một bản mô tả use case sẽ có dạng bảng như sau:

Tên use case:	**ID:**	**Mức độ quan trọng:**
Tác nhân chính:	**Loại use case:**	
Mô tả ngắn gọn:		
Điều kiện khởi phát:		
Quan hệ:		
Luồng hoạt động:		
Thứ tự	*Thực hiện bởi*	*Hành động*
1	<Hệ thống hoặc tác nhân>	Hành động 1
.....		
n	<Hệ thống hoặc tác nhân>	Hành động n
Luồng hoạt động con:		

Giải thích thêm về luồng hoạt động: Luồng hoạt động thể hiện các bước mà hệ thống cần thực hiện. Trong luồng hoạt động có thể có những luồng hoạt động con – điều này xảy ra khi cần biểu diễn sự rẽ nhánh, những trường hợp đặc biệt, những ngoại lệ có thể xảy ra.

2.2.4 Mô tả luồng hoạt động

Luồng hoạt động được thể hiện trực quan hơn thông qua sơ đồ hoạt động. Luồng hoạt động cần thiết khi cần mô tả một quy trình nghiệp vụ, một use case hoặc một thao tác phức tạp.

Không nên nhầm lẫn giữa sơ đồ hoạt động với sơ đồ tuần tự. Bởi sơ đồ hoạt động tập trung vào mô tả nghiệp vụ, trong khi đó sơ đồ tuần tự thể hiện thứ tự gọi các phương thức, sự tương tác giữa các object.

Tham khảo các ký hiệu của sơ đồ hoạt động trong phần cùng tên của chương 1.

2.3 Phân tích tĩnh

Kết quả của quá trình phân tích là mô hình phân tích. *Mô hình phân tích* chính là cầu nối giữa *mô hình. Trong quá trình này, chúng ta sẽ các định các lớp (class) phân tích. Tuy nhiên những lớp này hoàn toàn có thể được thay đổi trong giai đoạn thiết kế.*

2.3.1 Xác định các lớp phân tích

Có thể nói rằng không có quy tắc chung nào cho việc xác định các lớp. Điều này đòi hỏi sự sáng tạo.

Các lớp được xác định có thể thay đổi dần dần trong quá trình phân tích và thiết kế. Kết quả cuối cùng chưa chắc đã là các lớp được xác định trong lần đầu. Vì vậy không nên quá sa đà vào việc xác định các lớp phân tích.

Một số cách xác định các lớp:

- Như chúng ta đã biết thì lập trình hướng đối tượng là một mô hình lập trình dựa trên khái niệm đối tượng – Các đối tượng trong đời sống thực tế có thể được ánh xạ thành các đối tượng trong lập trình. Như vậy, có thể xác định các lớp ban đầu bằng cách tìm ra những **danh từ** trong tài liệu.

- Mỗi lớp phải thực hiện những nhiệm vụ, tính năng cụ thể – ta có thể dựa vào *sơ đồ use case* để phát hiện các lớp.
- Ta còn có thể xác định lớp nhờ một số cách phân loại:
  - *Lớp khái niệm*: Lớp khái niệm mô tả những khái niệm quan trọng và những quan hệ trong một nhóm nào đó.
  - *Lớp sự kiện*: Lớp sự kiện lưu trữ những mốc thời gian, các sự kiện. Ví dụ: *đăng ký, đăng nhập, thanh toán, ...*
  - *Lớp tổ chức*: Tập hợp tài nguyên, phương tiện, những nhóm giúp xác định chức năng người dùng. Ví dụ: *đơn vị, bộ phận, phòng ban, ...*
  - *Lớp con người*: Thể hiện các vai trò khác nhau của con người. Ví dụ: *khách hàng, sinh viên, nhân viên, giáo viên, giám đốc, ...*
  - *Lớp vị trí* : Các lớp đại diện cho vị trí vật lý. Ví dụ: *tòa nhà, phòng, chi nhánh, công ty, ...*
  - *Lớp sự vật*: Là những đối tượng vật lý. Ví dụ: *xe, thiết bị, máy móc, ...*

2.3.2 Xác định quan hệ giữa các lớp

Nhắc lại (từ chương 1, phần *sơ đồ lớp*), quan hệ giữa các lớp (class) thuộc một trong các loại sau:

- Thừa kế/tổng quát hóa (inheritance, generalization).
- Kết hợp (association).
- Kết tập (aggregation).
- Hợp thành (composition).

Trong đó, với quan hệ kết hợp, kết tập và hợp thành ta cần chỉ ra bội số ($1-1$, $1-n$, $n-n$).

2.3.3 Lập sơ đồ lớp

Từ những điều trên, ta có thể xây dựng *sơ đồ lớp* thể hiện được quan hệ giữa các lớp. Xem về *sơ đồ lớp* tại chương 1.

2.3.4 Xác định thuộc tính lớp

2.4 Phân tích động

2.5 Xây dựng sơ đồ giao tiếp hoặc sơ đồ tuần tự

2.6 Gán phương thức cho các lớp

2.7 Thiết kế kiến trúc hệ thống

2.8