
CHƯƠNG II

KHÁI QUÁT VỀ UML VÀ

CÁC CÔNG CỤ PHÁT TRIỂN HỆ THỐNG

NỘI DUNG CHÍNH

2.1 Giới thiệu về UML

2.2 Các biểu đồ trong UML

2.3 Giới thiệu công cụ Rational Rose

2.1 Giới thiệu về UML

2.1.1 Lịch sử ra đời của UML

2.1.2 Đặc trưng và mục đích sử dụng

2.1.3 Các thành phần của ngôn ngữ UML

2.1.4 Một số khái niệm trong UML

2.1.5 Các phần tử của mô hình và quan hệ

2.1.1 Lịch sử ra đời của UML

- ❑ **Giai đoạn:** (1960s – 1970s)
 - Cobol, Fortran, C
 - Structured analysis and design technique
- ❑ **Giai đoạn:** (1980s - đầu 1990s)
 - Smalltalk, Ada, C⁺⁺, Visual Basic
 - Early generation – OO methods
- ❑ **Giai đoạn:** cuối 1990 (giai đoạn ra đời của UML).
 - Ngôn ngữ lập trình Java
 - UML (Unified Modelling Language)
 - Unified Process

2.1.2 Đặc trưng và mục đích sử dụng 1

- UML là ngôn ngữ mô hình hóa tổng quát được xây dựng để đặc tả, phát triển và viết tài liệu cho các khía cạnh (view- hướng nhìn) trong phát triển phần mềm hướng đối tượng.
- UML giúp người phát triển hiểu rõ và ra quyết định liên quan đến phần mềm cần xây dựng.
- UML bao gồm tập các khái niệm, ký hiệu, các biểu đồ và hướng dẫn.

2.1.2 Đặc trưng và mục đích sử dụng 2

- UML hỗ trợ xây dựng hệ thống hướng đối tượng dựa trên việc nắm bắt khía cạnh cấu trúc tĩnh và các hành vi động của hệ thống.
- **Các cấu trúc tĩnh** định nghĩa các kiểu đối tượng quan trọng của hệ thống, nhằm cài đặt và chỉ ra mối quan hệ giữa các đối tượng.
- **Các hành vi động** (dynamic behavior) định nghĩa các hoạt động của các đối tượng theo thời gian và tương tác giữa các đối tượng hướng tới đích.

2.1.2 Đặc trưng và mục đích sử dụng 3

- Mô hình hóa các hệ thống sử dụng các khái niệm hướng đối tượng.
- Thiết lập sự liên hệ từ nhận thức của con người đến các sự kiện cần mô hình hóa.
- Giải quyết vấn đề mức độ thừa kế trong các hệ thống phức tạp với nhiều ràng buộc khác nhau.
- Tạo một ngôn ngữ mô hình hóa có thể sử dụng được bởi người và máy.

2.1.3 Các thành phần của ngôn ngữ UML

- **Hướng nhìn (view):**

Chỉ ra những khía cạnh khác nhau của hệ thống cần phải được mô hình hóa. ví dụ: *khía cạnh cấu trúc hệ thống, khía cạnh động, ...*

- **Biểu đồ (diagram):**

Là các hình vẽ miêu tả nội dung trong một hướng nhìn. UML có tất cả 9 loại biểu đồ khác nhau được sử dụng trong những sự kết hợp khác nhau để cung cấp tất cả các hướng nhìn của một hệ thống.

- **Phần tử mô hình hóa (model element):**

Các khái niệm được sử dụng trong các biểu đồ được gọi là các phần tử mô hình, thể hiện các khái niệm hướng đối tượng quen thuộc. ví dụ: *lớp, đối tượng, thông điệp, ...*

2.1.4 Một số khái niệm trong UML

2.1.4.1 Khái niệm mô hình

2.1.4.2 Kiến trúc hệ thống

2.1.4.3 Các hướng nhìn

2.1.4.1 Khái niệm mô hình

- Mô hình là một biểu diễn của sự vật hay một tập các sự vật trong một lĩnh vực áp dụng nào đó theo một cách khác.
- Mô hình nhằm nắm bắt các khía cạnh quan trọng của sự vật, bỏ qua các khía cạnh không quan trọng và biểu diễn theo một tập ký hiệu và qui tắc nào đó.
- Các mô hình thường được xây dựng sao cho có thể vẽ được các thành phần biểu diễn dựa trên tập ký hiệu và qui tắc đã cho.

Mục đích của việc sử dụng mô hình

- Nắm bắt chính xác yêu cầu và tri thức miền mà hệ thống cần phát triển.
- Thể hiện tư duy về thiết kế hệ thống.
- Trợ giúp ra quyết định thiết kế dựa trên việc phân tích yêu cầu.
- Tổ chức, tìm kiếm, lọc, kiểm tra và sửa đổi thông tin về các hệ thống lớn.
- Làm chủ được các hệ thống phức tạp.

Các thành phần trong một mô hình

- **Ngữ nghĩa và biểu diễn:** ngữ nghĩa là nhằm đưa ra ý nghĩa, bản chất và các tính chất của tập ký hiệu. Biểu diễn là phương pháp thể hiện mô hình theo cách sao cho có thể nhìn thấy được.
- **Ngữ cảnh:** mô tả tổ chức bên trong, cách sử dụng mô hình trong tiến trình phần mềm.

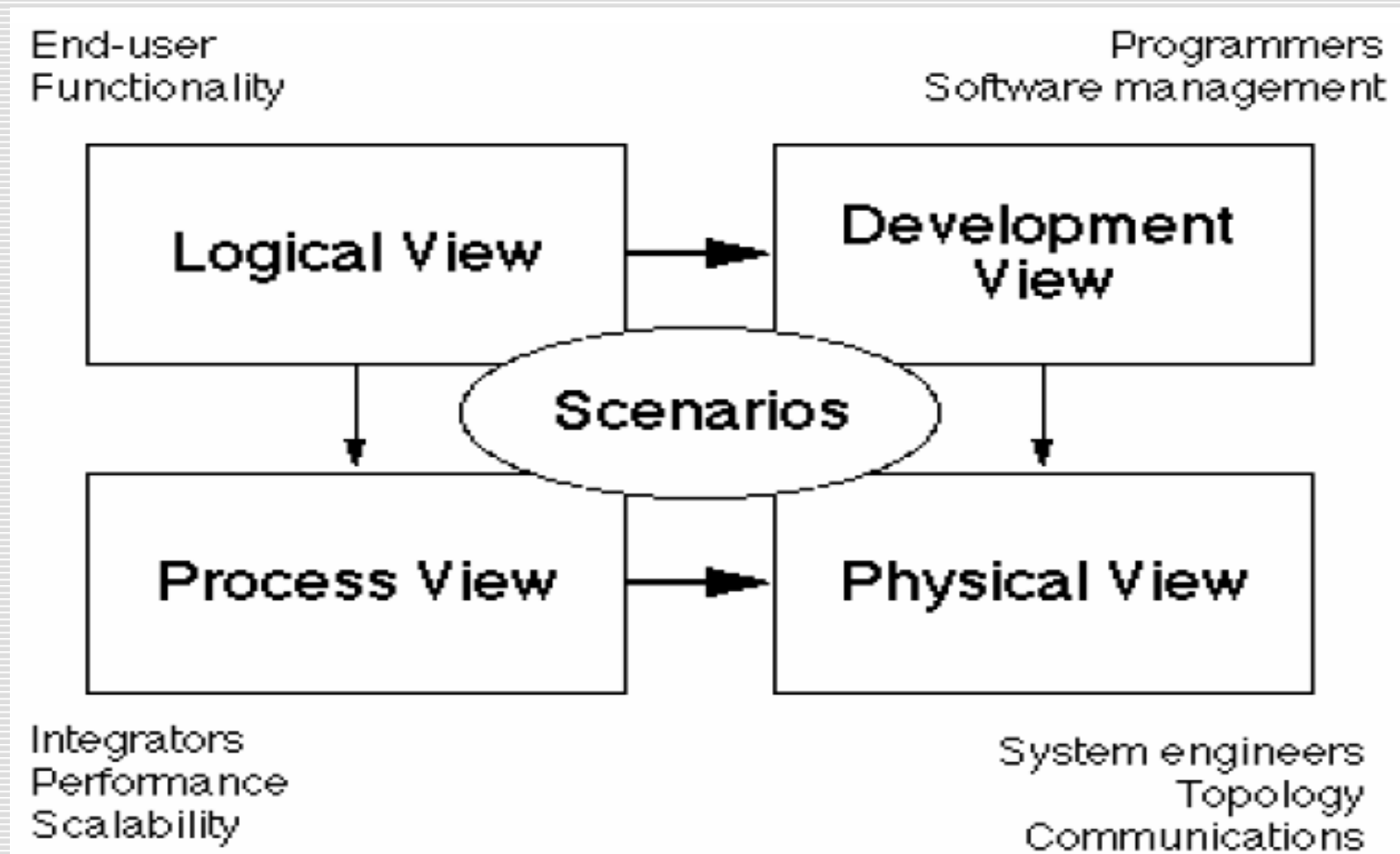
2.1.4.2 Kiến trúc hệ thống

- Kiến trúc hệ thống là trừu tượng hóa các khía cạnh quan trọng nhất của hệ thống.
- Cung cấp khung trong đó thiết kế được xây dựng.
- Mô tả tầm cỡ, sức mạnh của hệ thống, thu thập các user case quan trọng nhất và các yêu cầu ứng dụng.
- Thể hiện phần mềm sẽ được tổ chức như thế nào và cung cấp các giao thức trao đổi dữ liệu và giao tiếp giữa các modul.
- Là vật phẩm quan trọng nhất, được sử dụng để quản lý các hướng nhìn (view) khác nhau và điều khiển hệ thống tăng dần và lặp trong suốt chu kỳ sống.

KTHT là tập hợp các quyết định về

- Tổ chức của HT phần mềm
- Lựa chọn các phần tử cấu trúc và giao diện cho hệ thống.
- Hành vi của chúng thể hiện trong hợp tác giữa các phần tử.
- Tổ hợp các phần tử cấu trúc và hành vi vào hệ con lớn hơn.

Mô hình kiến trúc hệ thống



2.1.4.3 Các hướng nhìn (views)

- **Trong UML có 4 hướng nhìn cơ bản nhất**
 - Hướng nhìn user case (user case view)
 - Hướng nhìn logic (logic view)
 - Hướng nhìn thành phần (component view)
 - Hướng nhìn song song (concurrency view)

Hướng nhìn user case (user case view)

- Miêu tả chức năng của hệ thống sẽ phải cung cấp, do được tác nhân (actor) bên ngoài mong đợi. (*actor là thực thể tương tác với hệ thống, như người dùng, hoặc hệ thống khác*).
- Là hướng nhìn cho khách hàng, nhà thiết kế, nhà phát triển và người thử nghiệm.
- Mang tính trung tâm, vì nó tạo ra sự thúc đẩy và phát triển các hướng nhìn khác.

Hướng nhìn logic (logical view)

- Miêu tả phương thức mà các chức năng của hệ thống sẽ được cung cấp (hướng nhìn này chủ yếu cho nhà thiết kế và nhà phát triển).
- Nhìn vào bên trong của hệ thống, nó miêu tả cả cấu trúc tĩnh (lớp, đối tượng, quan hệ) cũng như sự tương tác động sẽ xảy ra khi các đối tượng gửi thông điệp cho nhau để cung cấp chức năng đã định sẵn.
- Định nghĩa các thuộc tính như: trường tồn (persistency), song song (concurrency), giao diện (interface) và cấu trúc nội tại của hệ thống.

Hướng nhìn thành phần (component view)

- Miêu tả việc thực thi của các modul cũng như sự phụ thuộc giữa chúng. (thường được sử dụng cho nhà phát triển và thường bao gồm nhiều biểu đồ thành phần).
- Thành phần ở đây là các modul thuộc nhiều loại khác nhau, sẽ được chỉ ra trong biểu đồ cùng với cấu trúc cũng như sự phụ thuộc của chúng.

Hướng nhìn song song (concurrency view)

- Nhằm tới việc chia hệ thống thành các qui trình (process) và các bộ xử lý (processor). Ở khía cạnh này, vốn là một thuộc tính phi chức năng của hệ thống, cho phép chúng ta sử dụng một cách hữu hiệu các nguồn tài nguyên, thực thi song song, cũng như xử lý các sự kiện không đồng bộ từ môi trường.
- Ngoài ra hướng nhìn này còn quan tâm đến vấn đề giao tiếp và đồng bộ hóa các tiến trình đó.
- Hướng nhìn song song giành cho nhà phát triển và người tích hợp hệ thống.

Bảng các hướng nhìn trong UML1

❑ Khía cạnh cấu trúc hệ thống

Hướng nhìn	Các biểu đồ	Các khái niệm
Hướng nhìn tĩnh (static view)	Biểu đồ lớp	Lớp, liên hệ, kế thừa, phụ thuộc, giao diện
Hướng nhìn user case (user case view)	Biểu đồ user case	User case, tác nhân, liên hệ, extend, include,...
Hướng nhìn cài đặt (implementation view)	Biểu đồ thành phần	Thành phần, giao diện, quan hệ phụ thuộc,...
Hướng nhìn triển khai (deployment view)	Biểu đồ triển khai	Node, thành phần, quan hệ phụ thuộc, vị trí (location).

Bảng các hướng nhìn trong UML2

❑ Khía cạnh động

Hướng nhìn	Các biểu đồ	Các khái niệm
Hướng nhìn máy trạng thái (state machine view)	Biểu đồ trạng thái	Trạng thái, sự kiện, chuyển tiếp, kết hợp, đồng bộ
Hướng nhìn hoạt động (activity view)	Biểu đồ hoạt động	Trạng thái, sự kiện, chuyển tiếp, kết hợp, đồng bộ
Hướng nhìn tương tác (interaction view)	Biểu đồ tuần tự	Tương tác, đối tượng, thông điệp, kích hoạt,...
	Biểu đồ cộng tác	Cộng tác, vai trò cộng tác, thông điệp,...

Bảng các hướng nhìn trong UML3

❑ Khía cạnh quản lý mô hình

Hướng nhìn

Các biểu đồ

Các khái niệm

Hướng nhìn quản lý mô hình Biểu đồ lớp

Gói, hệ thống, mô hình

2.1.5 Các phần tử của mô hình và quan hệ

2.1.5.1 Phần tử cấu trúc

2.1.5.2 Phần tử hành vi

2.1.5.3 Phần tử nhóm

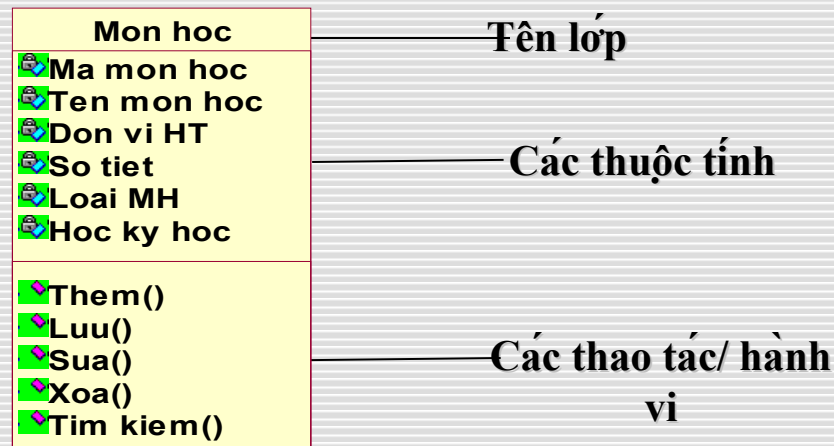
2.1.5.4 Chú thích

2.1.5.5 Các dạng quan hệ trong UML

2.1.5.1 Phần tử cấu trúc

□ Lớp (class):

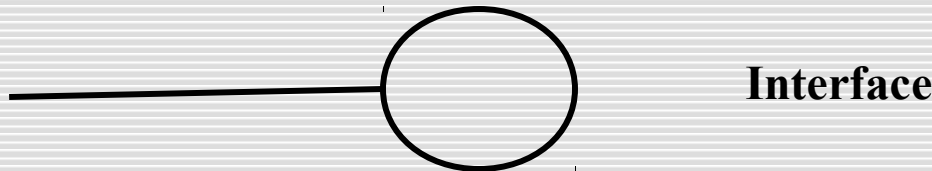
- Là mô tả các đối tượng có chung thuộc tính, thao tác, quan hệ và ngữ nghĩa. ví dụ lớp môn học.



2.1.5.1 Phần tử cấu trúc

□ Giao diện (interface):

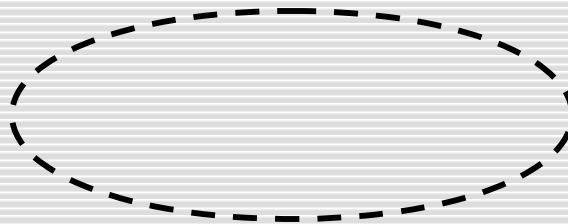
- Là tập hợp các thao tác làm dịch vụ của lớp hay thành phần.
- Giao diện biểu diễn toàn bộ hay một phần hành vi của lớp, nó định nghĩa tập đặc tả thao tác chứ không định nghĩa cài đặt của chúng.
- Giao diện thường không đứng một mình mà được gắn vào lớp hay thành phần thực hiện gd.



2.1.5.1 Phần tử cấu trúc

□ Phần tử cộng tác (collaboration):

- Mô tả ngữ cảnh của tương tác, thể hiện một giải pháp thi hành bên trong hệ thống, bao gồm: các lớp, quan hệ và tương tác giữa chúng để đạt được một chức năng mong đợi từ user case.
- Ký pháp đồ họa:



2.1.5.1 Phần tử cấu trúc

□ Trường hợp sử dụng (user case):

- Mô tả tập trình tự các hành động mà hệ thống sẽ thực hiện mà hệ thống sẽ thực hiện để đạt được một kết quả cho tác nhân nào đó. (tác nhân là những gì bên ngoài tương tác với hệ thống).
- Ký pháp đồ họa:



2.1.5.1 Phần tử cấu trúc

□ Thành phần (component):

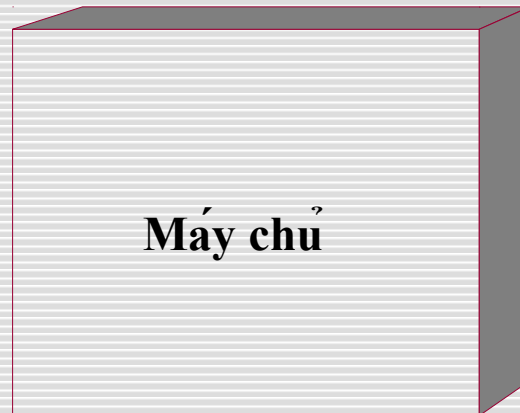
- Thành phần biểu diễn vật lý mã nguồn, các tệp nhị phân trong quá trình triển khai hệ thống.
- Ký pháp đồ họa:

`myfile.cpp`

2.1.5.1 Phần tử cấu trúc

□ Nút (node):

- Nút là thể hiện thành phần vật lý, tồn tại khi chương trình chạy và biểu diễn các tài nguyên tính toán.
- Ký pháp đồ họa:



2.1.5.2 Phần tử hành vi

□ Tương tác (interact):

- Là hành vi bao gồm tập các thông điệp trao đổi giữa các đối tượng trong ngữ cảnh cụ thể để thực hiện mục đích cụ thể.
- Ký pháp đồ họa:



2.1.5.2 Phần tử hành vi

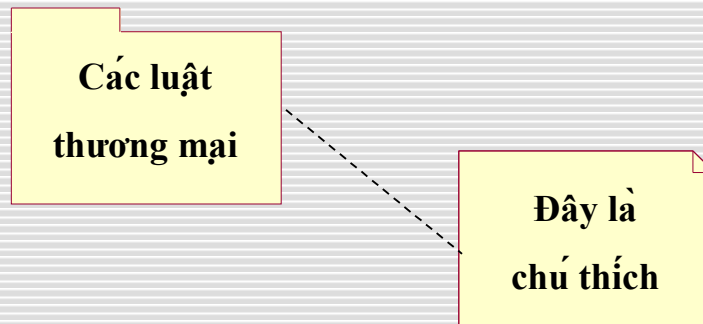
❑ Máy trạng thái (state machine):

- Máy trạng thái là hành vi chỉ ra trật tự các trạng thái mà đối tượng hay tương tác sẽ đi qua để đáp ứng sự kiện.
- Ký pháp đồ họa:



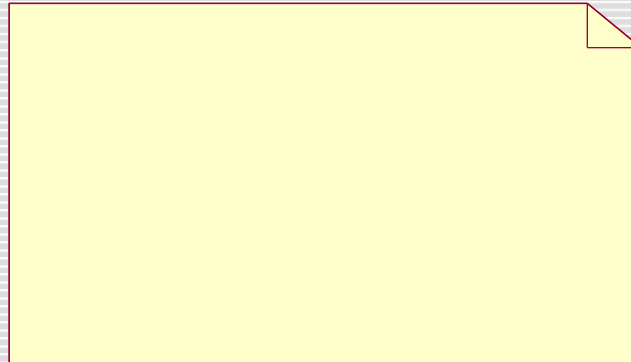
2.1.5.3 Phần tử nhóm (PTN)

- PTN là bộ phận tổ chức của mô hình UML. Chỉ có một phần tử thuộc nhóm này có tên là gói (package). Gói là cơ chế đa năng để tổ chức các phần tử vào nhóm. Các phần tử cấu trúc, hành vi, hay PTN có thể cho vào gói.
- Không giống thành phần, PTN hoàn toàn là một khái niệm, chúng chỉ tồn tại vào thời điểm phát triển hệ thống chứ không tồn tại vào thời điểm chạy chương trình.
- Ký pháp đồ họa:



2.1.5.4 Chú thích

- Phần tử chú thích (PTCT) là bộ phận chú thích của mô hình UML. Đó là lời giải thích áp dụng để mô tả các phần tử khác trong mô hình.
- PTCT được gọi là ghi chú (note).
- Ký pháp đồ họa:



2.1.5.5 Các dạng quan hệ trong UML

Quan hệ phụ thuộc (dependency):

- Là quan hệ ngữ nghĩa giữa hai phần tử trong đó thay đổi phần tử độc lập sẽ tác động đến ngữ nghĩa của phần tử phụ thuộc.
- Ký pháp đồ họa:

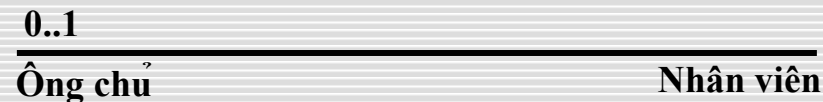
Quan hệ phụ thuộc



2.1.5.5 Các dạng quan hệ trong UML


❑ Quan hệ kết hợp (association):

- Là quan hệ cấu trúc để mô tả tập liên kết (một liên kết là kết nối giữa các đối tượng). Khi đối tượng của lớp này gửi / nhận thông điệp đến/ từ đến từ lớp kia thì ta gọi chúng là có quan hệ kết hợp.
- Ký pháp đồ họa:

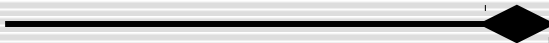


2.1.5.5 Các dạng quan hệ trong UML

❑ Tụ hợp (aggregation):


- Là một dạng đặc biệt của kết hợp, nó biểu diễn quan hệ giữa cấu trúc và bộ phận.
- Ký pháp đồ họa: 

❑ Hợp thành (composition):


- Một dạng đặc biệt của tập hợp là hợp thành, trong đó nếu như đối tượng toàn thể bị hủy bỏ thì các đối tượng bộ phận của nó cũng bị hủy bỏ theo.
- Ký pháp đồ họa: 

2.1.5.5 Các dạng quan hệ trong UML

□ Khái quát hóa (generalization):

- Khái quát hóa là quan hệ đặc biệt hóa/ khái quát hóa mà trong đó đối tượng cụ thể sẽ kế thừa các thuộc tính và phương pháp của đối tượng tổng quát.
- Ký pháp đồ họa: 

□ Hiện thực hóa (realization):

- Hiện thực hóa là quan hệ ngữ nghĩa giữa giao diện và lớp (hay thành phần) hiện thực lớp; giữa UC và hợp tác hiện thực user case.
- Ký pháp đồ họa: 

2.2 Các biểu đồ trong UML

2.2.1 Biểu đồ user case

2.2.2 Biểu đồ lớp (class diagram)

2.2.3 Biểu đồ trạng thái (state diagram)

2.2.4 Biểu đồ trình tự (sequency diagram)

2.2.5 Biểu đồ cộng tác (collaboration diagram)

2.2.6 Biểu đồ hoạt động (activity diagram)

2.2.7 Biểu đồ thành phần (component diagram)

2.2.8 Biểu đồ triển khai hệ thống (deployment diagram)

2.2.1 Biểu đồ user case

□ Ý nghĩa:

- Biểu diễn sơ đồ chức năng của hệ thống.
- Biểu đồ UC chỉ ra sự tương tác giữa tác nhân và hệ thống.
- Mỗi UC mô tả một chức năng của hệ thống cần phải xét từ quan điểm người sử dụng thông qua các kịch bản (scenario).
- Một biểu đồ UC là một tập hợp các Actor, các UC và các mối quan hệ giữa chúng.

2.2.1 Biểu đồ user case 2



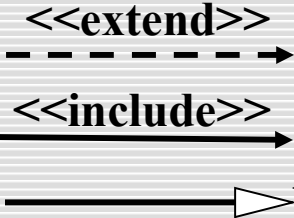

□ Tập ký hiệu trong biểu đồ user case :

- **Hệ thống (system):** với vai trò là thành phần của biểu đồ user case, hệ thống biểu diễn ranh giới bên trong và bên ngoài của một chu thể phần mềm chúng ta đang xây dựng.
- **Tác nhân (actor):** là người dùng của hệ thống, một actor có thể là một người dùng thực hoặc các hệ thống máy tính khác giữ vai trò nào đó trong hoạt động của hệ thống.
- **Các user case:** đây là thành phần cơ bản trong biểu đồ user case, các uc được biểu diễn bởi các hình elip.

2.2.1 Biểu đồ user case 3

- **Mối quan hệ giữa các user case:**
 - + **Include:** user case này sử dụng lại chức năng của uc kia.
 - + **Extend:** user case này mở rộng từ uc kia bằng cách thêm vào một chức năng cụ thể.
 - + **Generalization:** user case này được kế thừa các chức năng từ uc kia.

Các phần tử mô hình trong biểu đồ user case

PHẦN TỬ MÔ HÌNH	Ý NGHĨA	CÁCH BIỂU DIỄN	KÝ HIỆU TRONG BIỂU ĐỒ
User case	Biểu diễn một chức năng xác định của HT	Hình elip chứa tên của các user case	
Tác nhân (actor)	Là lớp đối tượng bên ngoài hệ thống tương tác trực tiếp với các UC	Biểu diễn bởi một lớp kiểu actor (hình người tượng trưng)	
Mối quan hệ giữa các UC	Tùy từng dạng quan hệ	Extend và include có dạng các mũi tên đứt nét. Generalization có dạng mũi tên tam giác	
Biên của hệ thống	Tách biệt phần bên trong và bên ngoài hệ thống	Được biểu diễn bởi một hình chữ nhật rỗng	

Country	Year	Value
Algeria	2000	0.00
Algeria	2001	0.00
Algeria	2002	0.00
Algeria	2003	0.00
Algeria	2004	0.00
Algeria	2005	0.00
Algeria	2006	0.00
Algeria	2007	0.00
Algeria	2008	0.00
Algeria	2009	0.00
Algeria	2010	0.00
Algeria	2011	0.00
Algeria	2012	0.00
Algeria	2013	0.00
Algeria	2014	0.00
Algeria	2015	0.00
Algeria	2016	0.00
Algeria	2017	0.00
Algeria	2018	0.00
Algeria	2019	0.00
Algeria	2020	0.00
Algeria	2021	0.00
Algeria	2022	0.00
Algeria	2023	0.00
Algeria	2024	0.00
Algeria	2025	0.00
Algeria	2026	0.00
Algeria	2027	0.00
Algeria	2028	0.00
Algeria	2029	0.00
Algeria	2030	0.00
Algeria	2031	0.00
Algeria	2032	0.00
Algeria	2033	0.00
Algeria	2034	0.00
Algeria	2035	0.00
Algeria	2036	0.00
Algeria	2037	0.00
Algeria	2038	0.00
Algeria	2039	0.00
Algeria	2040	0.00
Algeria	2041	0.00
Algeria	2042	0.00
Algeria	2043	0.00
Algeria	2044	0.00
Algeria	2045	0.00
Algeria	2046	0.00
Algeria	2047	0.00
Algeria	2048	0.00
Algeria	2049	0.00
Algeria	2050	0.00
Algeria	2051	0.00
Algeria	2052	0.00
Algeria	2053	0.00
Algeria	2054	0.00
Algeria	2055	0.00
Algeria	2056	0.00
Algeria	2057	0.00
Algeria	2058	0.00
Algeria	2059	0.00
Algeria	2060	0.00
Algeria	2061	0.00
Algeria	2062	0.00
Algeria	2063	0.00
Algeria	2064	0.00
Algeria	2065	0.00
Algeria	2066	0.00
Algeria	2067	0.00
Algeria	2068	0.00
Algeria	2069	0.00
Algeria	2070	0.00
Algeria	2071	0.00
Algeria	2072	0.00
Algeria	2073	0.00
Algeria	2074	0.00
Algeria	2075	0.00
Algeria	2076	0.00
Algeria	2077	0.00
Algeria	2078	0.00
Algeria	2079	0.00
Algeria	2080	0.00
Algeria	2081	0.00
Algeria	2082	0.00
Algeria	2083	0.00
Algeria	2084	0.00
Algeria	2085	0.00
Algeria	2086	0.00
Algeria	2087	0.00
Algeria	2088	0.00
Algeria	2089	0.00
Algeria	2090	0.00
Algeria	2091	0.00
Algeria	2092	0.00
Algeria	2093	0.00
Algeria	2094	0.00
Algeria	2095	0.00
Algeria	2096	0.00
Algeria	2097	0.00
Algeria	2098	0.00
Algeria	2099	0.00
Algeria	2100	0.00
Algeria	2101	0.00
Algeria	2102	0.00
Algeria	2103	0.00
Algeria	2104	0.00
Algeria	2105	0.00
Algeria	2106	0.00
Algeria	2107	0.00
Algeria	2108	0.00
Algeria	2109	0.00
Algeria	2110	0.00
Algeria	2111	0.00
Algeria	2112	

2.2.2 Biểu đồ lớp 1

□ Ý nghĩa:

- Trong hướng đối tượng, một nhóm đối tượng có chung một số thuộc tính và phương thức tạo thành một lớp.
- Mỗi tương tác giữa các đối tượng trong hệ thống sẽ được biểu diễn thông qua mối quan hệ giữa các lớp.
- Các lớp, bao gồm cả thuộc tính và phương thức cùng với các mối quan hệ sẽ tạo thành một biểu đồ lớp
- Biểu đồ lớp là một biểu đồ dạng mô hình tĩnh, nhằm mô tả hướng nhìn tĩnh về một hệ thống bằng các khái niệm lớp, các thuộc tính của lớp và quan hệ giữa chúng.

2.2.2 Biểu đồ lớp 2

❑ Tập ký hiệu UML cho biểu đồ lớp:

- **Ký hiệu lớp (class):** trong UML mỗi lớp được biểu diễn bởi hình chữ nhật gồm 3 phần: **tên lớp**, **các thuộc tính** và **các phương thức**.
- **Thuộc tính (attribute):** các thuộc tính trong biểu đồ lớp được biểu diễn theo cấu trúc chung như sau:

phạm vi_tên: kiểu_số_đối_tượng = mặc_định (gia_trị_giới_hạn)

✓ Trong đó:

- **Phạm vi:** cho biết phạm vi truy cập của thuộc tính, có 3 kiểu xác định thuộc tính phổ biến là:

2.2.2 Biểu đồ lớp 3

- + : thuộc tính kiểu **public**
- # : thuộc tính kiểu **protected**
- : thuộc tính kiểu **private**
- ~ : thuộc tính được phép truy cập tới từ các lớp trong cùng package.
- **Tên:** là xâu ký tự biểu diễn tên thuộc tính.
- **Kiểu:** là kiểu dữ liệu của thuộc tính.
- **Số_đối_tượng:** chỉ ra số đối tượng khai báo cho thuộc tính.
- **Mặc_định:** là giá trị khởi đầu mặc định của thuộc tính.
- **Giá_trị_giới_hạn:** là giới hạn các giá trị của thuộc tính.

2.2.2 Biểu đồ lớp 3

- Phương thức (method):

phạm vi tên (danh sách tham số): kiểu trả lại {kiểu phương thức}

✓ Trong đó:

– **Visibility**: biểu diễn phạm vi cho phương thức. Giống như đối với thuộc tính, có 3 dạng kiểu xác định cơ bản cho phương thức là:

+ : phương thức kiểu **public**

: phương thức kiểu **protected**

- : phương thức kiểu **private**

~ : phương thức được phép truy cập tới từ các lớp trong cùng package.

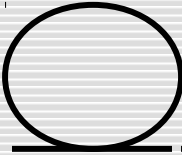
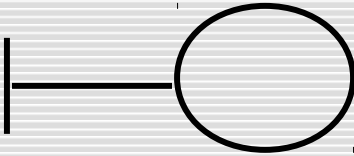
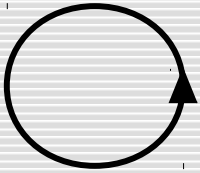
2.2.2 Biểu đồ lớp 4

- **Tên:** là xâu ký tự xác định tên của phương thức
- **Kiểu trả lại:** chỉ ra kiểu giá trị trả về của phương thức
- **Danh sách tham số:** biểu diễn danh sách các tham số trong khai báo của phương thức. Mỗi tham số được biểu diễn dưới dạng chung: tên tham số: *kiểu giá trị = giá trị mặc định.*
- **Kiểu phương thức:** không bắt buộc, cho biết kiểu phương thức. Phương thức có thể nhận 1 trong 2 giá trị:
 - + **abstract:** phương thức trừu tượng
 - + **query :** phương thức kiểu truy vấn

Các kiểu lớp trong UML

- **Lớp thực thể:** là đại diện cho các thực thể chứa thông tin về các đối tượng xác định nào đó.
- **Lớp biên (lớp giao diện):** là lớp nằm giữa ranh giới giữa hệ thống với môi trường bên ngoài, thực hiện vai trò nhận yêu cầu trực tiếp từ các tác nhân và chuyển các yêu cầu đó cho các lớp bên trong hệ thống.
- **Lớp điều khiển:** thực hiện các chức năng điều khiển hoạt động của hệ thống ứng với các chức năng cụ thể nào đó với một nhóm các lớp biên hoặc lớp thực thể xác định.

Bảng các kiểu lớp trong UML

Stt	Kiểu lớp	Ký hiệu UML
1	Lớp thực thể	
2	Lớp biên (lớp giao diện)	
3	Lớp điều khiển	

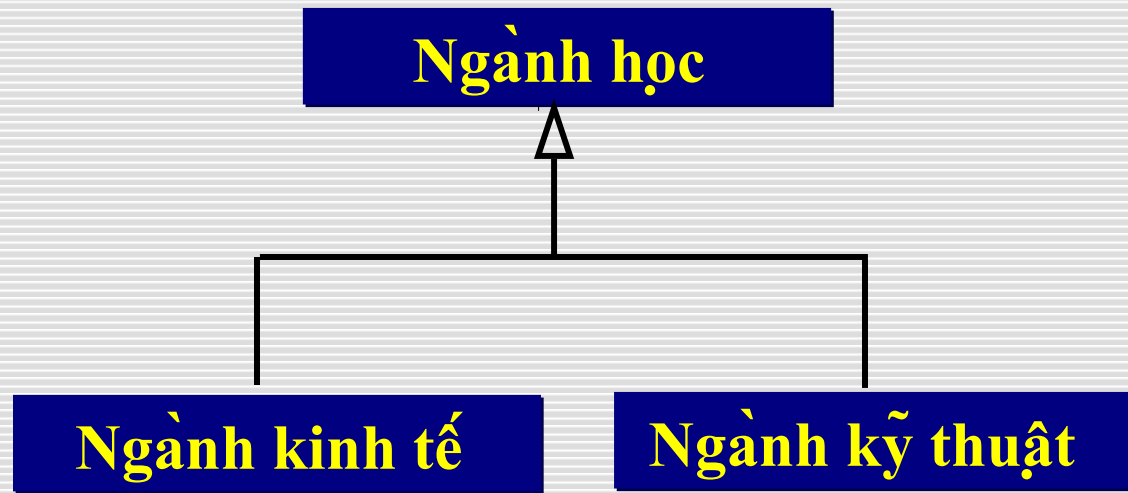
Các mối quan hệ trong biểu đồ lớp 1

- Quan hệ kết hợp (association):
 - Một kết hợp là một sự kết nối giữa các lớp, cũng có nghĩa là sự kết nối giữa các đối tượng của các lớp này.
 - Ví dụ quan hệ giữa **khách hàng** và **sản phẩm**.



Các mối quan hệ trong biểu đồ lớp 2

- **Khái quát hóa (generalization):**
 - Khái quát hóa là mối quan hệ giữa một lớp có các đặc trưng mang tính khái quát cao hơn và một lớp có tính chất đặc biệt hơn.
 - Ví dụ quan hệ giữa **ngành học** và **chi tiết của ngành học**.



Các mối quan hệ trong biểu đồ lớp 3

- Quan hệ cộng hợp (Aggregation):
 - Là dạng quan hệ mô tả lớp A là một phần của lớp B và lớp A có thể tồn tại độc lập.
 - Ví dụ quan hệ giữa **hóa đơn** và **khách hàng**.



Các mối quan hệ trong biểu đồ lớp 4

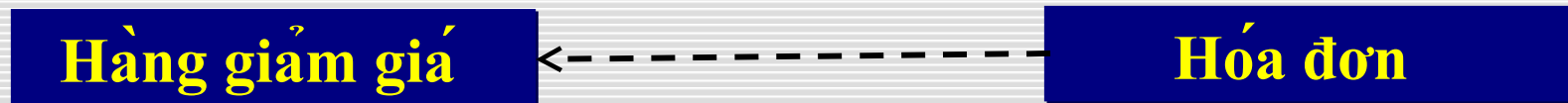
- **Quan hệ gộp (Composition):**

- Một quan hệ gộp biểu diễn một quan hệ kiểu tập thể và bộ phận. Lớp A có quan hệ gộp với lớp B nếu lớp A là một phần của lớp B và sự tồn tại của đối tượng B điều khiển sự tồn tại của đối tượng A.
- Ví dụ quan hệ giữa **địa chỉ** và **khách hàng**.



Các mối quan hệ trong biểu đồ lớp 5

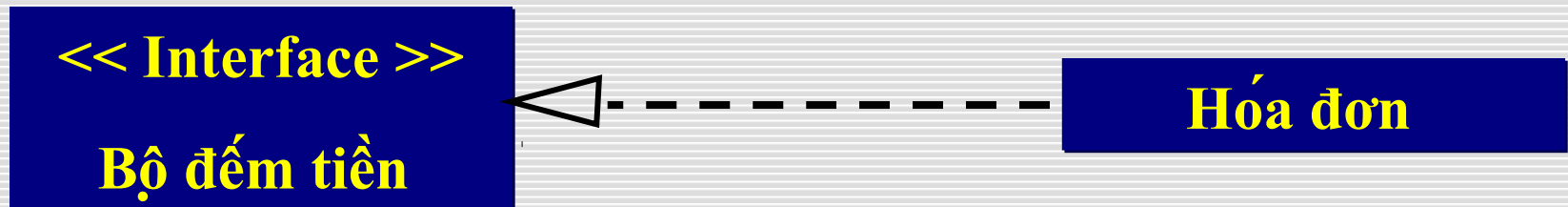
- **Quan hệ phụ thuộc (Dependency):**
 - Phụ thuộc là mối quan hệ giữa hai lớp đối tượng: một lớp đối tượng A có tính độc lập và một lớp đối tượng B phụ thuộc vào A, nếu thay đổi A sẽ ảnh hưởng đến lớp phụ thuộc B.
 - Ví dụ quan hệ giữa **Hàng giảm giá** và **Hóa đơn**.



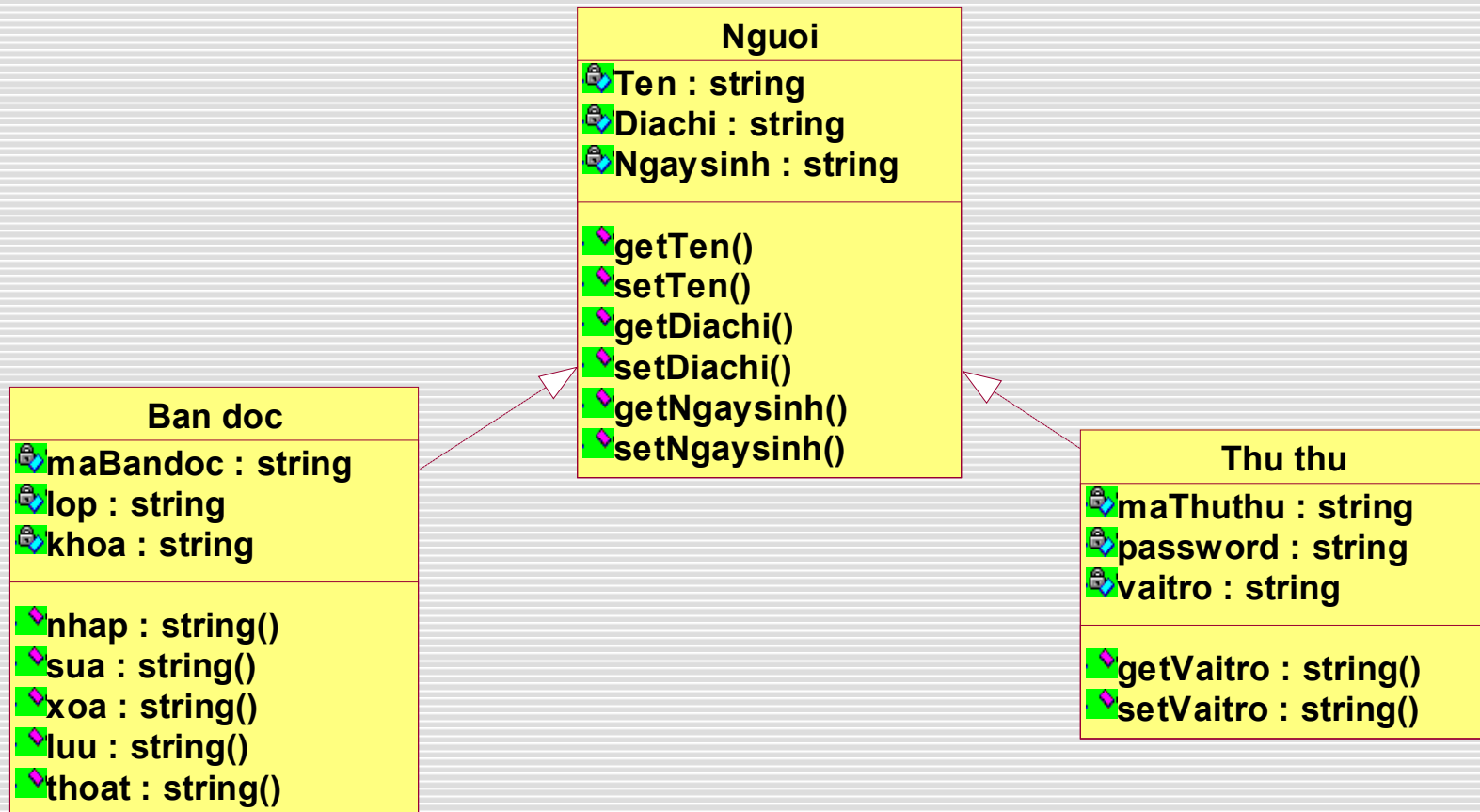
Các mối quan hệ trong biểu đồ lớp 6

- Quan hệ thực thi (Realization):

- Biểu diễn mối quan hệ ngữ nghĩa giữa các thành phần của biểu đồ lớp, trong đó một thành phần mô tả một công việc dạng hợp đồng và thành phần còn lại thực hiện hợp đồng đó. Thông thường thực hiện hợp đồng có thể là các giao diện.
- Ví dụ quan hệ giữa **Bộ đếm tiền** và **Hóa đơn**.



Ví dụ biểu đồ lớp trong hệ thống thư viện



2.2.3 Biểu đồ trạng thái

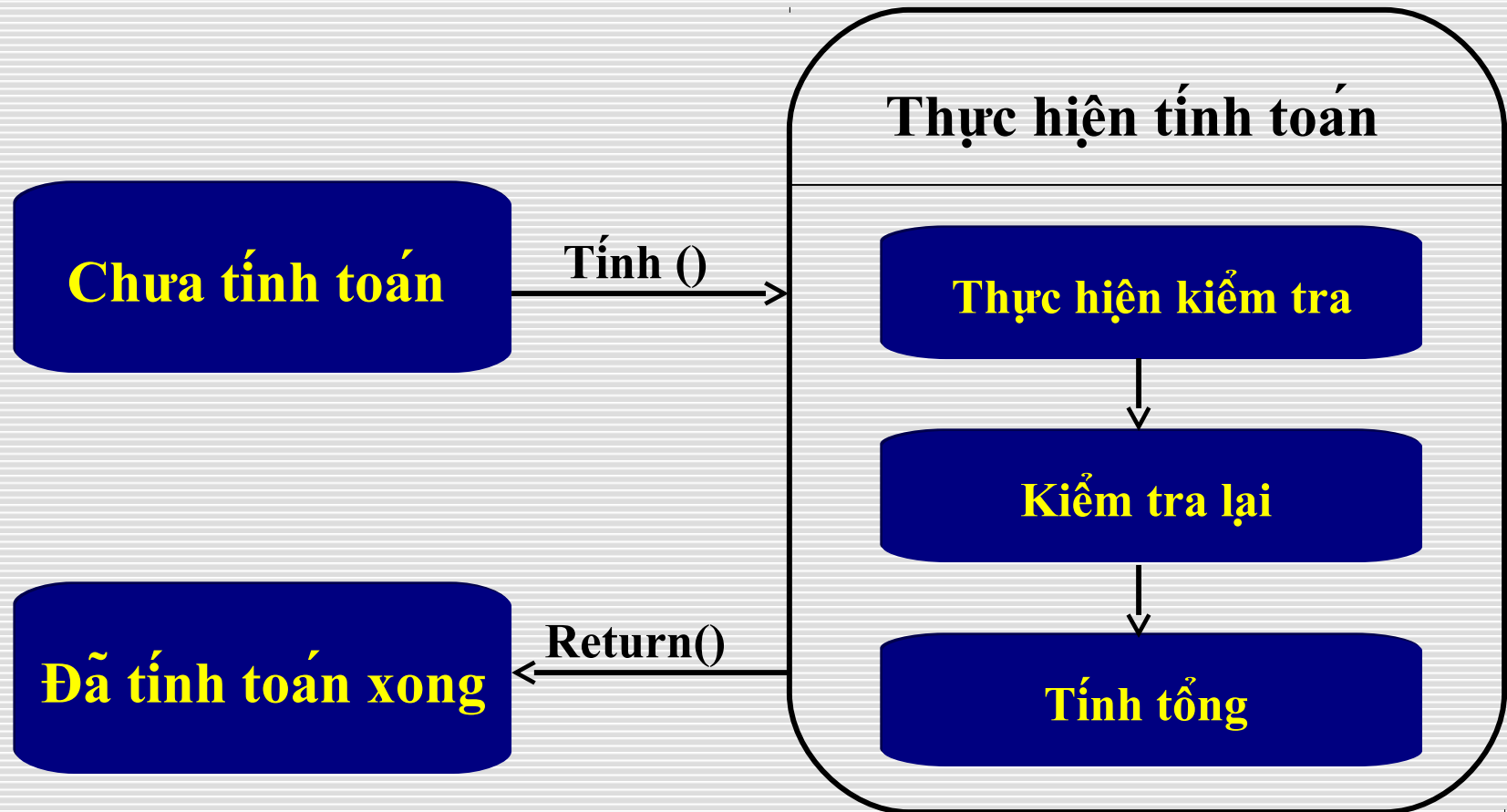
- **Ý nghĩa:**

- Biểu đồ trạng thái được sử dụng để biểu diễn các trạng thái và sự chuyển tiếp giữa các trạng thái của các đối tượng trong một lớp xác định. Thông thường, mỗi lớp sẽ có một biểu đồ trạng thái (trừ trường hợp là lớp không có đối tượng).
- Biểu đồ trạng thái biểu diễn dưới dạng máy trạng thái hữu hạn với các trạng thái và sự chuyển tiếp giữa các trạng thái đó. Có hai dạng biểu đồ trạng thái:
 - + Biểu đồ trạng thái cho một user case
 - + Biểu đồ trạng thái hệ thống mô tả tất cả các trạng thái của các đối tượng trong toàn bộ các hoạt động của hệ thống.

Tập ký hiệu trong biểu đồ 1

- **Trạng thái (state)**: bên trong các trạng thái có thể miêu tả các biến trạng thái hoặc các hành động (action) tương ứng với trạng thái đó.
- **Trạng thái con (substate)**: là một trạng thái chứa bên trong một trạng thái khác. Trạng thái có nhiều trạng thái con gọi là trạng thái tổ hợp.
- **Trạng thái khởi đầu (initial state)**: trạng thái đầu tiên khi kích hoạt đối tượng.
- **Trạng thái kết thúc (final state)**: kết thúc vòng đời đối tượng.

Ví dụ biểu đồ trạng thái có trạng thái con



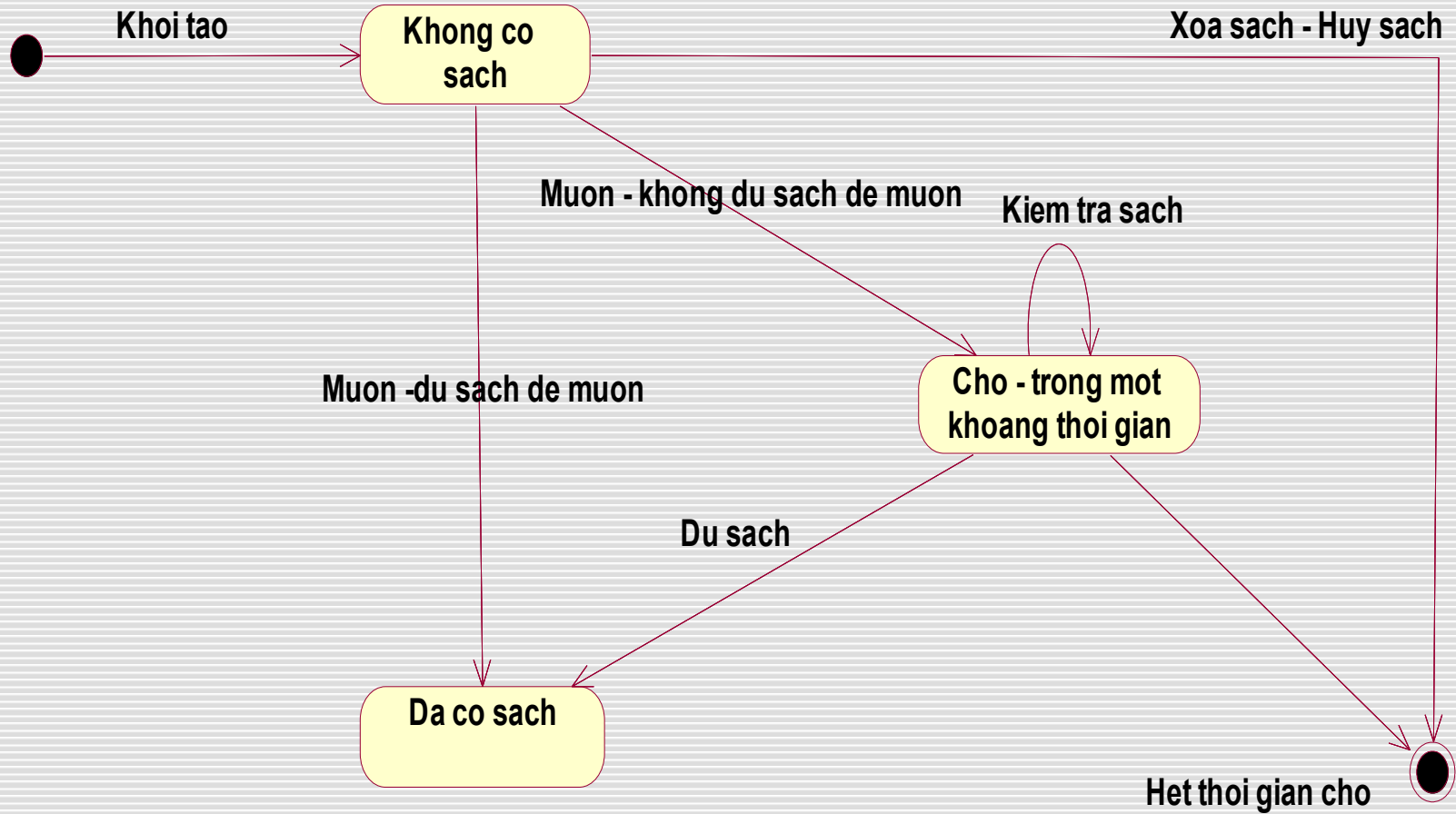
Tập ký hiệu trong biểu đồ trạng thái 2

- **Các chuyển tiếp (transition):** biểu diễn các chuyển đổi giữa các trạng thái.
- **Sự kiện (event):** sự kiện tác động gây ra sự chuyển đổi trạng thái. Mỗi sự kiện được đi kèm với các điều kiện (guard) và các hành động (action). Có 3 loại sự kiện:
 - **Sự kiện gọi (call event):** yêu cầu thực hiện một hành động, một phương thức.
 - **Sự kiện tín hiệu (signal event):** gửi thông điệp giữa các trạng thái.
 - **Sự kiện thời gian (time event):** biểu diễn quá trình chuyển tiếp theo thời gian, thường kèm theo từ mô tả thời gian cụ thể.

Các phần tử mô hình trong biểu đồ trạng thái

PHẦN TỬ MÔ HÌNH	Ý NGHĨA	BIỂU DIỄN	KÝ HIỆU TRONG BIỂU ĐỒ
Trạng thái	Biểu diễn một trạng thái của đối tượng trong vòng đời của đối tượng đó	Hình chữ nhật vòng ở góc, gồm 3 phần: tên, các biến, và các hoạt động.	
Trạng thái khởi đầu	Khởi đầu vòng đời của đối tượng	Hình tròn đặc	
Trạng thái kết thúc	Kết thúc vòng đời của đối tượng	Hai hình tròn lồng vào nhau	
Chuyển tiếp	Chuyển từ trạng thái này sang trạng thái khác	Mũi tên liền nét với tên gọi là biểu diễn của chuyển tiếp đó.	

Ví dụ biểu đồ trạng thái



2.2.4 Biểu đồ trình tự

- **Ý nghĩa:**

- Biểu diễn mối quan hệ giữa các đối tượng; giữa các đối tượng và tác nhân theo thứ tự thời gian. Biểu đồ trình tự nhấn mạnh thứ tự thực hiện của các tương tác.

- **Tập ký hiệu trong biểu đồ trình tự :**

- **Các đối tượng (object):** được biểu diễn bởi các hình chữ nhật, bên trong là tên của đối tượng.
- **Các message:** được biểu diễn bằng các mũi tên hướng từ đối tượng gửi sang đối tượng nhận.

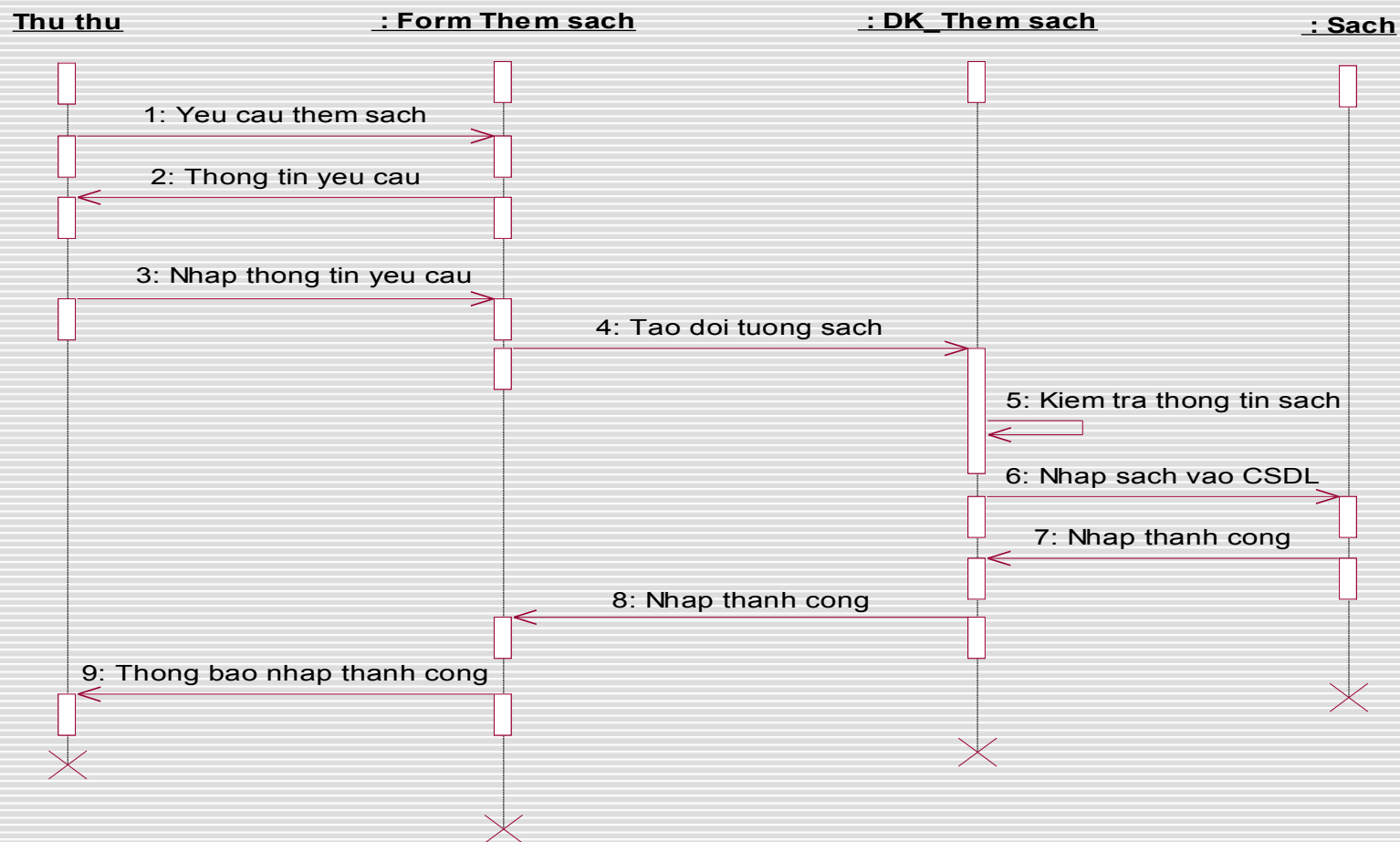
Bảng các loại message

Stt	Loại message	Mô tả	Biểu diễn
1	Gọi (call)	Mô tả một lời gọi từ đối tượng này đến đối tượng kia	
2	Trả về (return)	Trả về giá trị ứng với lời gọi	
3	Gửi (send)	Gửi một tín hiệu tới một đối tượng	
4	Tạo (create)	Tạo một đối tượng	
5	Hủy (destroy)	Hủy một đối tượng	

2.2.4 Biểu đồ trình tự

- **Đường lifeline:** là một đường kẻ nối dài phía đối tượng, mô tả quá trình của đối tượng trong tương tác thuộc biểu đồ.
- **Chú thích:** biểu đồ trình tự cũng có thể có chú thích để người đọc dễ dàng hiểu được nội dung của biểu đồ đó.

Ví dụ biểu đồ trình tự



2.2.5 Biểu đồ cộng tác

- **Ý nghĩa:**

- Là biểu đồ tương tác biểu diễn mối quan hệ giữa các đối tượng; giữa các đối tượng và tác nhân, nhấn mạnh đến vai trò của các đối tượng trong tương tác.

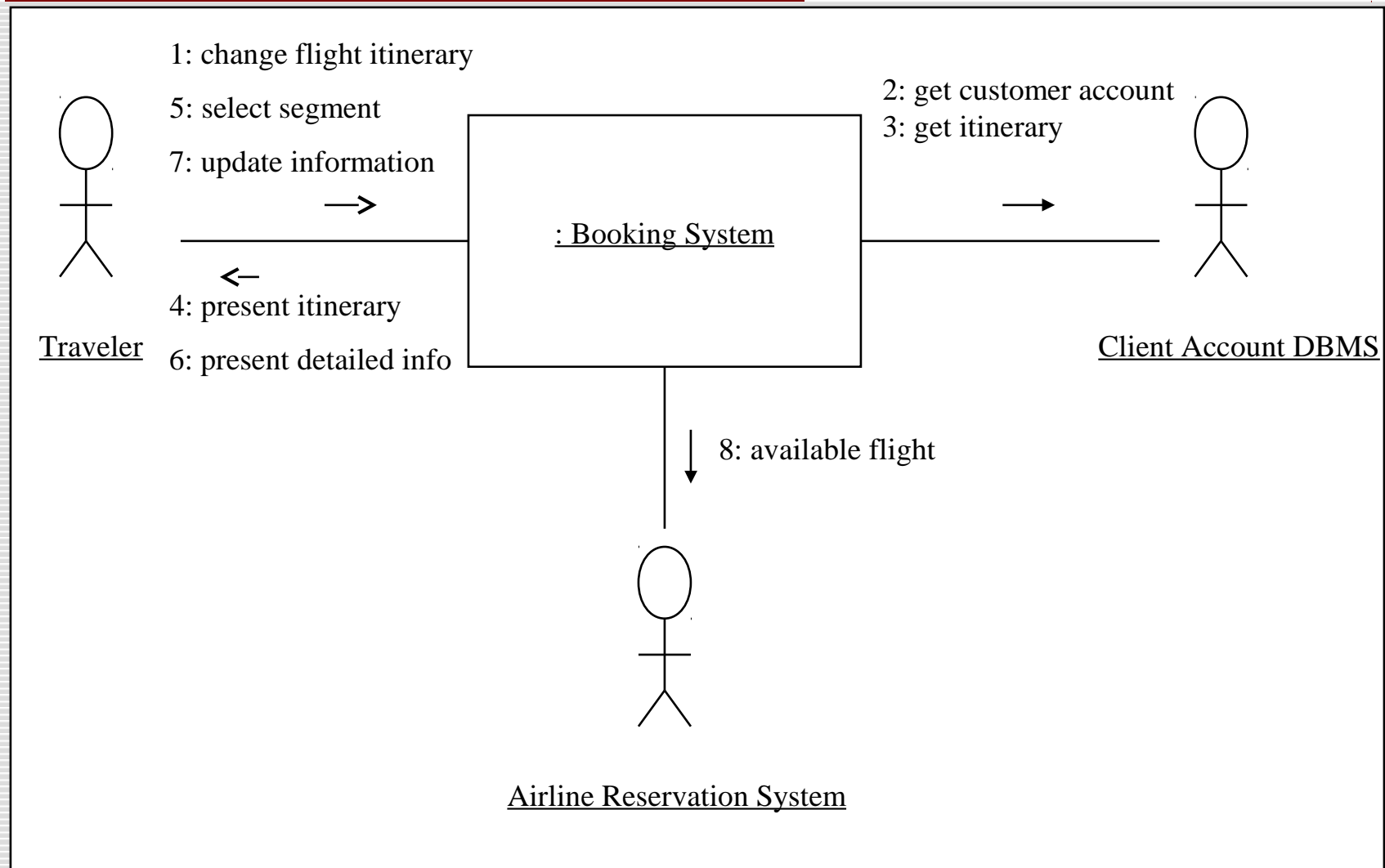
- **Tập ký hiệu trong biểu đồ**

- **Các đối tượng:** biểu diễn bởi hình chữ nhật, bên trong là tên đối tượng

- **Các liên kết:** giữa hai đối tượng có tương tác sẽ có một liên kết nối 2 đối tượng đó.

- **Các message:** được biểu diễn bằng các mũi tên hướng từ đối tượng gửi sang đối tượng nhận bên cạnh liên kết giữa hai đối tượng.

Ví dụ biểu đồ cộng tác



2.2.6 Biểu đồ hoạt động

- **Ý nghĩa:**

- Biểu diễn các hoạt động và sự đồng bộ, chuyển tiếp các hoạt động của hệ thống trong một lớp hoặc kết hợp giữa các lớp với nhau trong một chức năng cụ thể. Ngoài ra biểu đồ hoạt động được sử dụng để:
- Xác định các hành vi phải thực hiện trong phạm vi một phương thức.
- Xác định công việc của một đối tượng.
- Chỉ ra một nhóm các hành động liên quan của các đối tượng được thực hiện như thế nào và chúng sẽ ảnh hưởng đến các đối tượng nằm xung quanh.

2.2.6 Biểu đồ hoạt động

- Tập ký hiệu UML

- **Hoạt động (activity):** là một qui trình được định nghĩa rõ ràng, có thể được thực hiện bởi hàm hoặc một nhóm đối tượng. Hoạt động được thể hiện bằng hình chữ nhật tròn cạnh.



- **Thanh đồng bộ hóa (synchronisation bar):** cho phép ta mở ra hoặc đóng lại các nhánh chạy song song của tiến trình.



2.2.6 Biểu đồ hoạt động

- **Điều kiện (guard condition):** các biểu thức logic có giá trị đúng hoặc sai. Điều kiện được thể hiện trong ngoặc vuông.

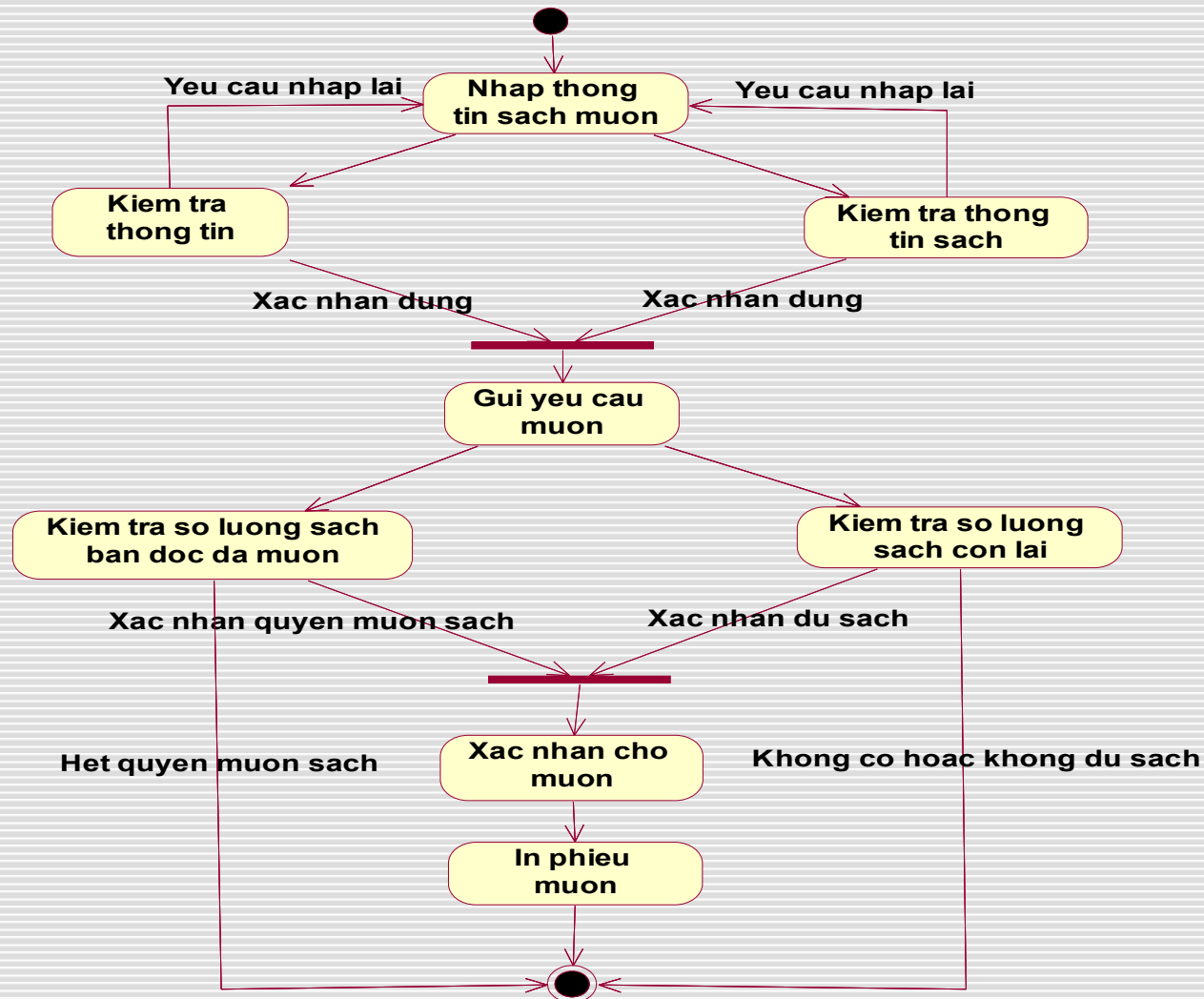


- **Các luồng (swimlane):** mỗi biểu đồ động có thể biểu diễn sự phối hợp hoạt động trong nhiều lớp khác nhau. Khi đó mỗi lớp được phân tách bởi một luồng riêng biệt.

Các phần tử trong biểu đồ hoạt động

Phần tử mô hình	Ý nghĩa	Ký hiệu
Hoạt động	Mô tả một hoạt động	
Trạng thái khởi đầu		
Trạng thái kết thúc		
Thanh đồng bộ hóa ngang	Mô tả thanh đồng bộ nằm ngang	
Thanh đồng bộ hóa dọc	Mô tả thanh đồng bộ dọc	
Chuyển tiếp		
Quyết định	Mô tả lựa chọn điều kiện	
Các luồng	Phân tách các đối tượng khác nhau tồn tại trong biểu đồ	Phân tách bởi đường kẻ dọc từ trên xuống dưới biểu đồ.

Ví dụ biểu đồ hoạt động



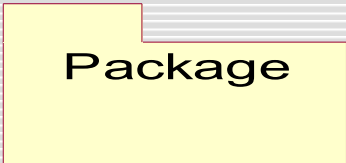


2.2.7 Biểu đồ thành phần

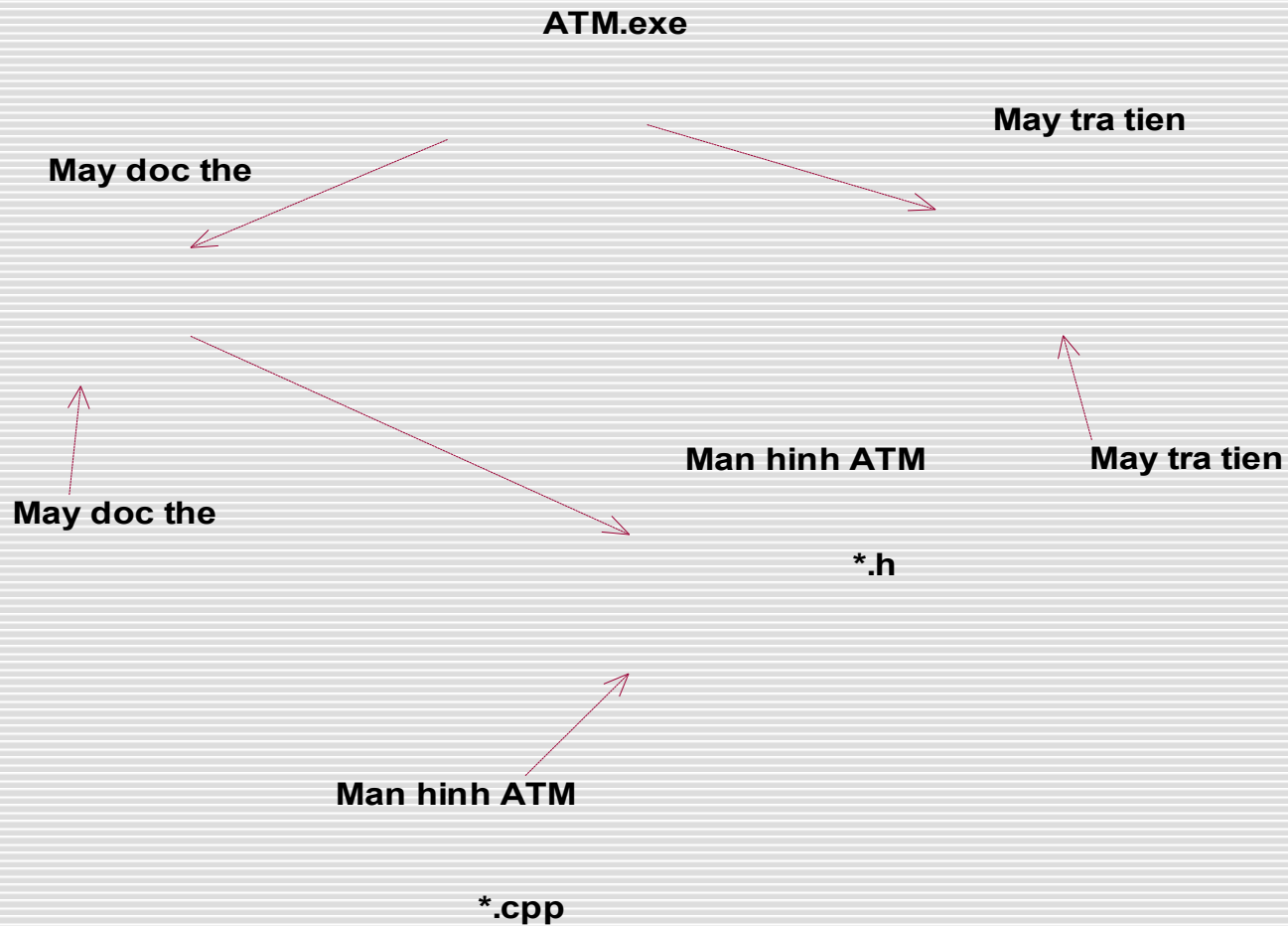
- **Ý nghĩa:**

- Được sử dụng để biểu diễn các thành phần phần mềm cấu thành hệ thống.
- Mỗi thành phần có thể xem như một phần mềm nhỏ hơn, cung cấp một khối dạng hộp đen trong quá trình xây dựng phần mềm lớn.
- Các thành phần có thể là các gói được xây dựng cho quá trình phát triển hệ thống.
- **Ví dụ:** Java Bean, các gói thư viện liên kết động, lớp và các thư viện chức năng.

Các ký hiệu trong biểu đồ thành phần

Phần tử mô hình	Ý nghĩa	Ký hiệu
Thành phần	Mô tả thành phần của biểu đồ	Component
Giao tiếp	Mô tả giao tiếp với mỗi thành phần	 Giao tiếp
Mối quan hệ phụ thuộc giữa các thành phần	Mô tả quan hệ giữa các thành phần	
Gói (package)	Được sử dụng để nhóm một số thành phần lại với nhau	

Ví dụ biểu đồ thành phần



2.2.8 Biểu đồ triển khai hệ thống

- Ý nghĩa:

- Biểu đồ triển khai hệ thống biểu diễn kiến trúc cài đặt và triển khai hệ thống dưới dạng các nodes và các mối quan hệ giữa các node.
- Thông thường các node được kết nối với nhau thông qua các liên kết truyền thông như các kết nối mạng, liên kết TCP-IP, microware....và được đánh số thứ tự theo thời gian tương tự như biểu đồ cộng tác.

Tập ký hiệu UML trong biểu đồ triển khai

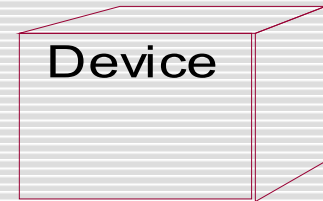
Phần tử mô hình

Ý nghĩa

Ký hiệu

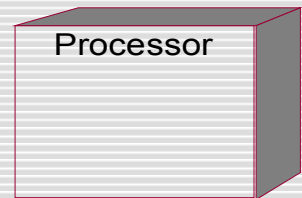
Các node (thiết bị)

Biểu diễn các thành phần không có bộ xử lý trong biểu đồ triển khai hệ thống



Các bộ xử lý

Biểu diễn các thành phần có bộ xử lý trong biểu đồ triển khai hệ thống

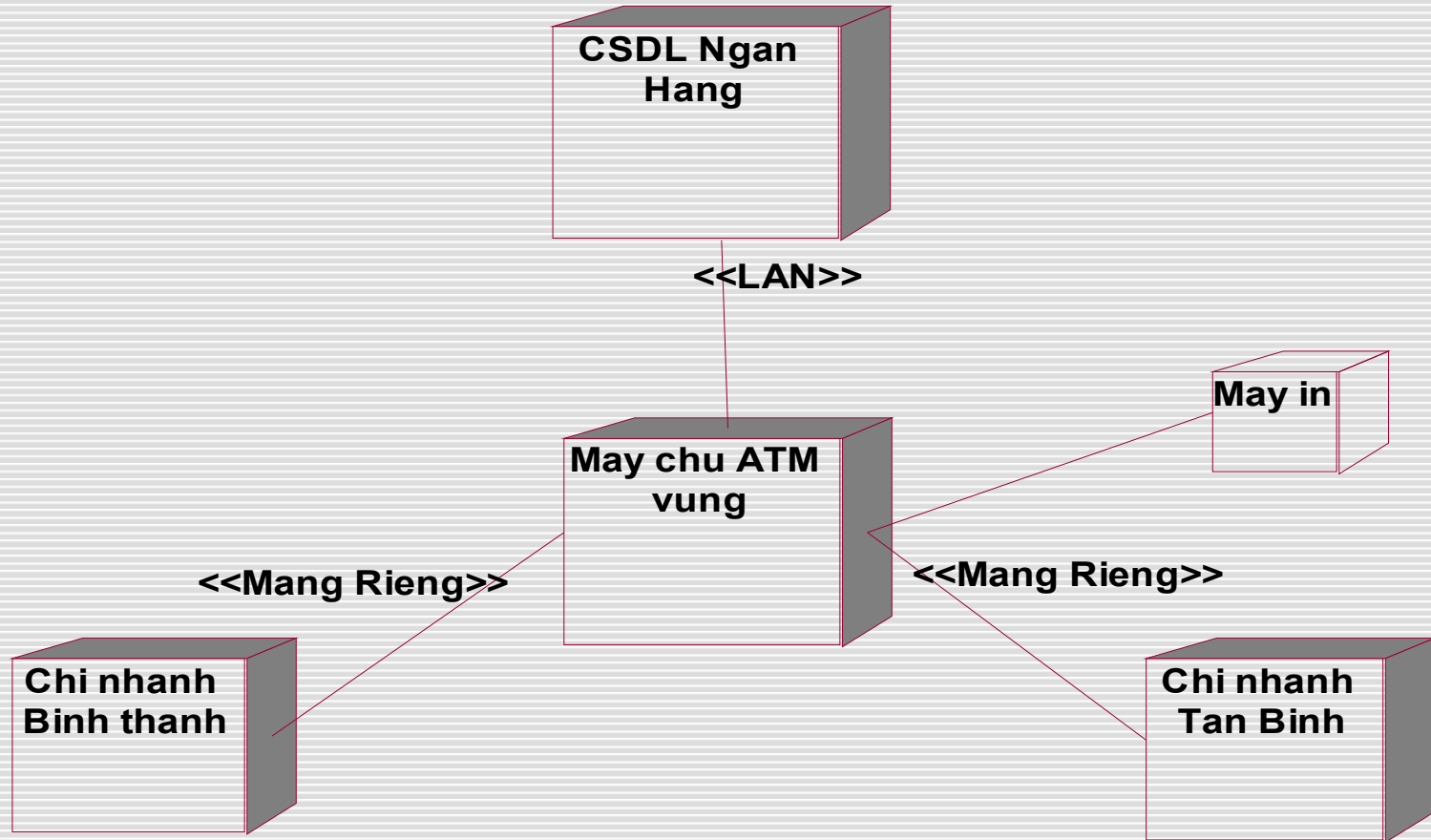


Các liên kết truyền thông

Nối các thành phần của biểu đồ triển khai hệ thống. Thường mô tả một giao thức truyền thông cụ thể



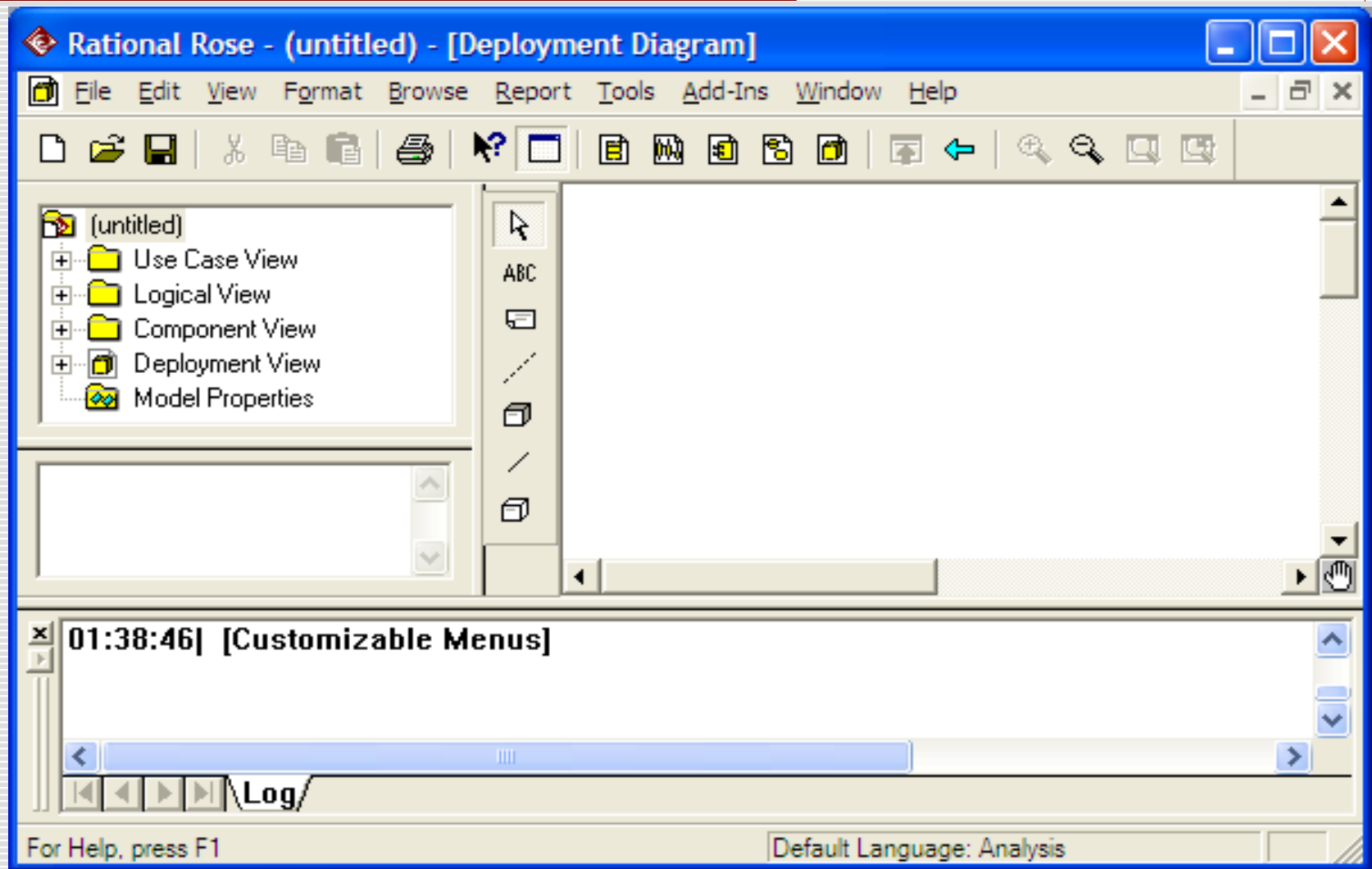
Ví dụ biểu đồ triển khai



2.3 Giới thiệu công cụ Rational Rose

- là một bộ công cụ được sử dụng cho phát triển hệ thống phần mềm hướng đối tượng theo ngôn ngữ mô hình hóa UML.
- Cho phép tạo ra, quan sát, sửa đổi và quản lý các biểu đồ.
- Tập ký hiệu trong Rational Rose cung cấp thống nhất với các ký hiệu trong UML

Màn hình làm việc của Rational Rose



Các thành phần chính trong Rational Rose

- **User Case View:** xem xét khía cạnh chức năng của hệ thống nhìn từ phía các tác nhân bên ngoài.
- **Logic View:** xem xét quá trình phân tích và thiết kế logic của hệ thống để thực hiện các chức năng trong user case view.
- **Component View:** xem xét khía cạnh tổ chức hệ thống theo các thành phần và mối liên hệ giữa các thành phần.
- **Deployment View:** xem xét khía cạnh triển khai hệ thống theo các kiến trúc vật lý.

Câu hỏi ôn tập chương 2

1. UML ra đời từ ngôn ngữ và pp mô hình hóa nào ?
2. Hướng nhìn là gì ? UML có những hướng nhìn nào ?
Khái niệm hướng nhìn dùng để làm gì ?
3. Liệt kê các biểu đồ của UML và tập ký hiệu cho mỗi biểu đồ ?
4. Liệt kê các bước phát triển phần mềm hướng đối tượng sử dụng UML ?
5. Phân biệt mô hình tĩnh và mô hình động trong UML ?
6. Liệt kê và lấy ví dụ về các dạng quan hệ trong UML ?