

# Phân tích thiết kế hướng đối tượng

## Bài 1: Giới thiệu

**TS. Nguyễn Hiếu Cường**

Bộ môn CNPM, Khoa CNTT, Trường ĐH GTVT

Email: [cuonggt@gmail.com](mailto:cuonggt@gmail.com)

# Giới thiệu môn học

---

- Mục đích
  - Hiểu được quy trình cơ bản để phân tích, thiết kế hướng đối tượng và
  - Sử dụng UML để làm tài liệu phân tích thiết kế
- Đánh giá
  - Điểm Quá trình (tỷ lệ 40%)
  - Điểm Kết thúc học phần (tỷ lệ 60%)
    - Bài tập lớn (làm theo nhóm) + Vấn đáp (cá nhân)

# Tài liệu

---

- Tài liệu tham khảo
  - System Analysis and Design with UML (Dennis et al.)
  - Applying UML and Patterns (Larman)
  - ...
- Phần mềm
  - Diagrams.net, draw.io (online)
  - Rational ROSE, Visual Paradigm (\$\$\$)
  - StarUML, Violet (free)
  - ...

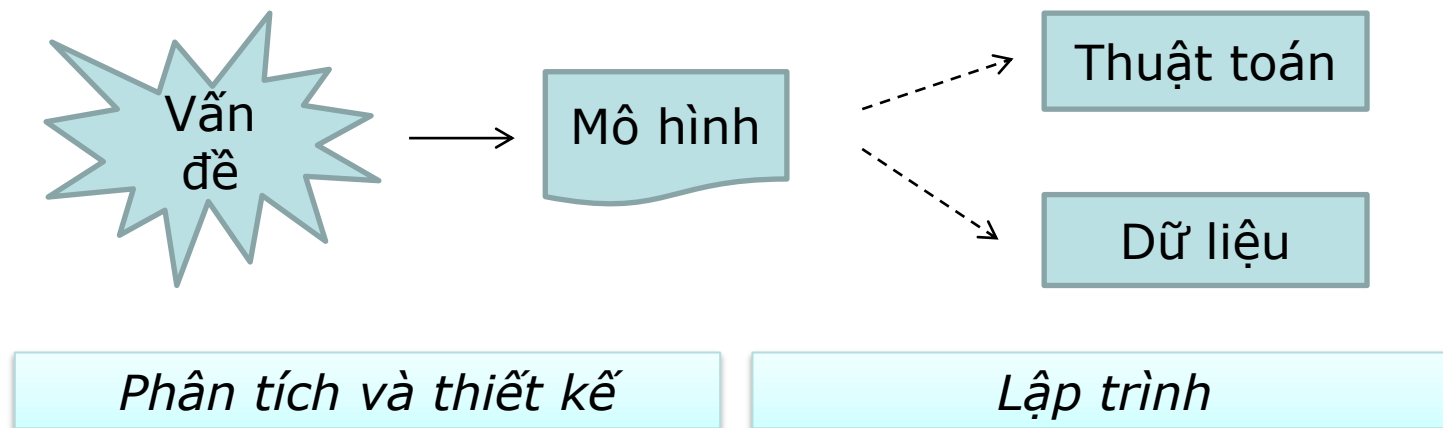
# Phát triển hệ thống

---

- Phát triển hệ thống
  - Quá trình xây dựng hệ thống từ những ý tưởng, khảo sát, phân tích, thiết kế đến cài đặt, kiểm thử và chuyển giao
  - Cần phải hiểu rõ hệ thống, làm rõ cách thức giải quyết
- Định hướng phát triển hệ thống
  - Hướng chức năng
  - Hướng đối tượng → Phân tích thiết kế hướng đối tượng (OOAD)

# Mô hình hóa

- Để giải quyết vấn đề cần
  - Phân tích và thiết kế = Mô hình hoá
  - Giải quyết vấn đề đã được mô hình hóa = Lập trình



# Ngôn ngữ mô hình hóa

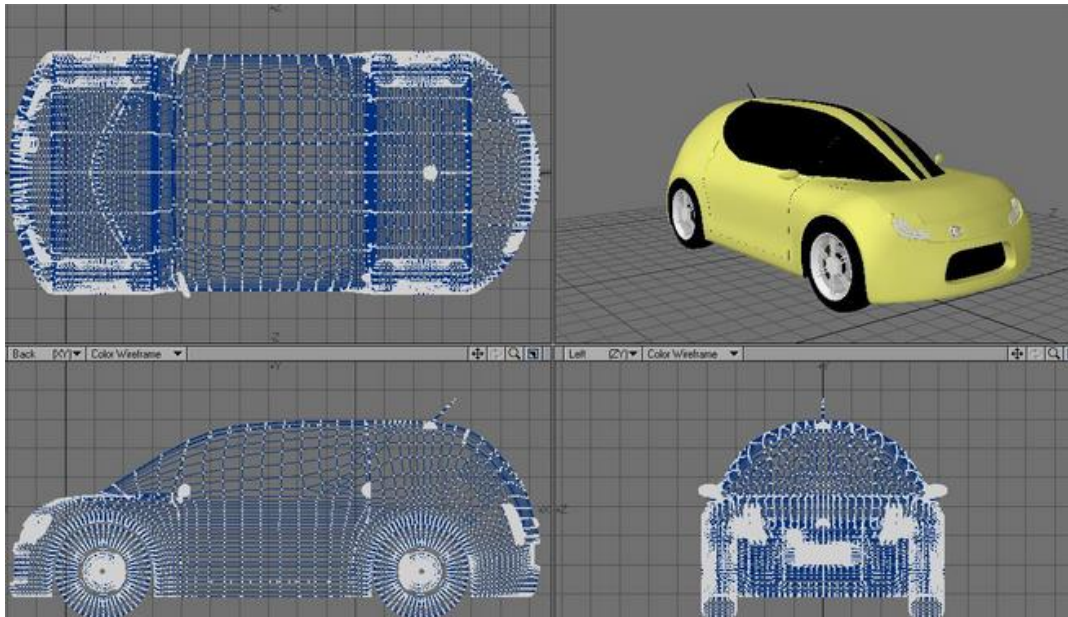
---

- Mô hình hóa
  - Mô phỏng hệ thống bằng ngôn ngữ mô hình hóa
  - Tập trung vào khía cạnh mình quan tâm
  - Cách thức thường dùng để xử lý vấn đề phức tạp
- Việc mô hình hoá cần có những hình thức biểu đạt
  - Văn bản
  - Hình vẽ, đồ thị, sơ đồ, ...



# Biểu diễn mô hình

- Mỗi đối tượng có thể được xem xét dưới nhiều góc độ



# Ngôn ngữ mô hình hóa thống nhất

---

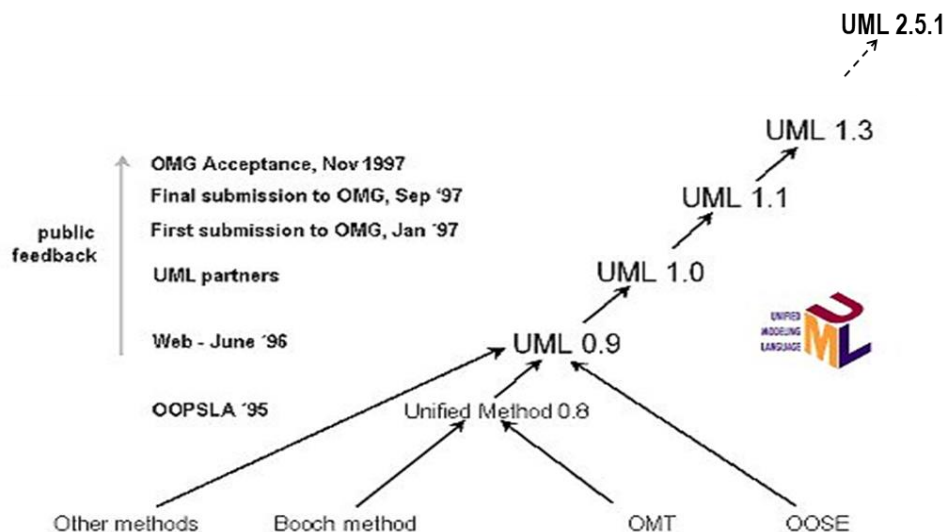
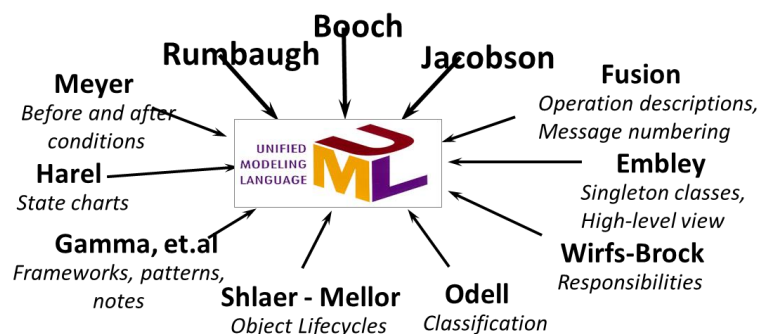
- UML (Unified Modeling Language)
  - Các nguyên tắc và ký hiệu đã chuẩn hóa
  - Biểu diễn và lưu trữ các mô hình
- Các đặc điểm của UML?
  - Phù hợp với mô hình hóa hướng đối tượng
  - Có thể mô hình trực quan, rõ ràng, chính xác
  - Làm tài liệu
  - ...





# Ngôn ngữ mô hình hóa thống nhất

- Các phương pháp mô hình hóa hướng đối tượng
  - Đã tồn tại khá nhiều phương pháp, với ký pháp khác nhau
  - Không thống nhất, khó trao đổi
  - Năm 1997 thì UML ra đời



# Tại sao cần UML

---

- UML = Unified Modeling Language, chuẩn hóa bởi OMG
- Không phải phương pháp luận, mà là ngôn ngữ mô hình hoá
- Lợi ích:
  - Giao tiếp giữa khách hàng – phân tích – lập trình
  - Giảm độ phức tạp hệ thống
  - Hỗ trợ thiết kế, bảo trì, tái sử dụng
- UML được dùng xuyên suốt vòng đời phần mềm:  
Yêu cầu → phân tích → thiết kế → triển khai → bảo trì

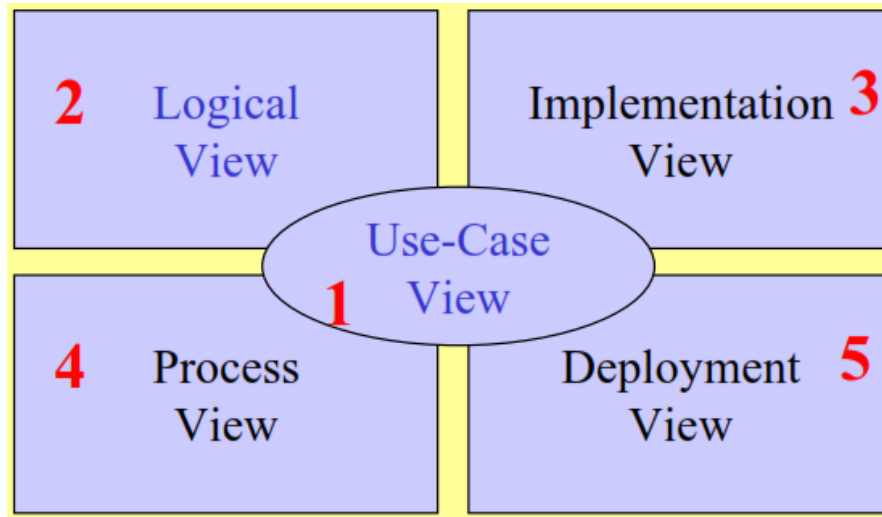
# Các nhóm sơ đồ

---

- Nhóm mô tả **chức năng** và yêu cầu
  - Use Case Diagram: ai dùng hệ thống, làm gì
- Nhóm mô tả **cấu trúc** tĩnh
  - Class Diagram: mô tả lớp, thuộc tính, quan hệ
  - Component Diagram (ở mức cao hơn, cho dự án lớn)
- Nhóm mô tả **hành vi** động
  - Sequence Diagram: luồng tương tác đối tượng
  - Activity Diagram: dòng công việc/quy trình
  - State Machine Diagram: vòng đời của một đối tượng

# Khái niệm về góc nhìn

- Có năm (4 + 1) góc nhìn/khung nhìn (view), mỗi góc nhìn dùng mô tả một khía cạnh của hệ thống



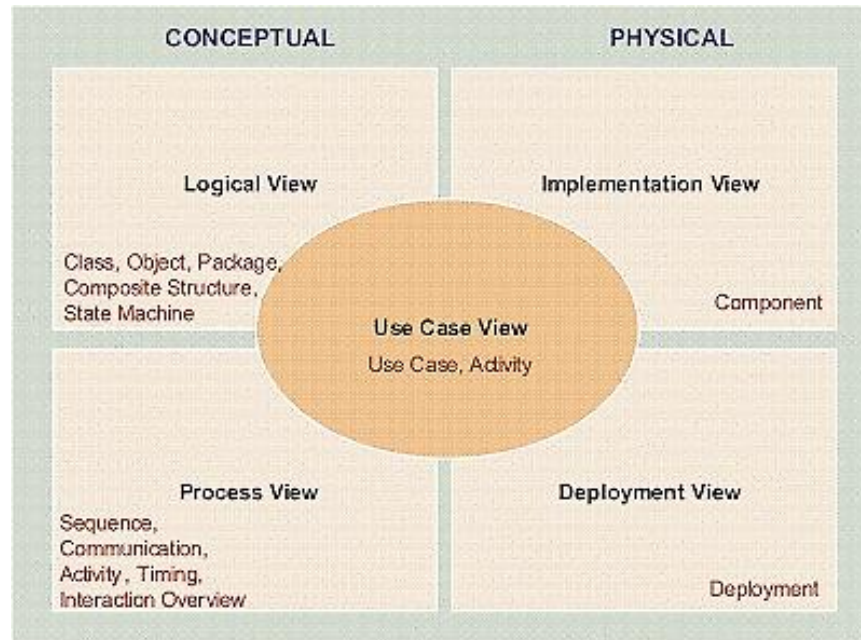
# Vai trò của góc nhìn

---

- Góc nhìn là tư duy tổ chức thiết kế
  - Mỗi góc nhìn nhấn mạnh một khía cạnh nhất định của hệ thống
  - Cách chia hệ thống theo mục đích: ai cần gì thì nhìn vào đâu
- Ví dụ trong hệ thống thư viện đơn giản
  - Người dùng quan tâm đến chức năng (→ Use Case View)
  - Người phân tích cần biết các lớp xử lý mượn sách (→ Logical View)
  - Lập trình viên cần tổ chức mã nguồn (→ Development View)

# Thể hiện các góc nhìn

- Mỗi góc nhìn được thể hiện bằng một hoặc một số loại sơ đồ



# Góc nhìn và các tên gọi khác nhau

Góc nhìn (View)	Tên gọi khác	Mục tiêu chính	Sơ đồ UML phổ biến
1. Use Case View	Scenarios View	Mô tả tương tác giữa người dùng (actor) và hệ thống	Use Case Diagram
2. Logical View	Design View, Static View	Mô tả cấu trúc lớp, thực thể, mối quan hệ giữa chúng	Class Diagram, Object Diagram
3. Process View	Dynamic View, Behavioral View	Mô tả hành vi động, luồng điều khiển, xử lý đồng thời	Sequence Diagram, Activity Diagram, State Machine Diagram
4. Development View	Implementation View	Mô tả tổ chức mã nguồn, cấu trúc các thành phần phần mềm	Component Diagram, Package Diagram
5. Physical View	Deployment View	Mô tả cách triển khai phần mềm lên phần cứng (hạ tầng vật lý hoặc đám mây)	Deployment Diagram

# Mô hình hóa với UML

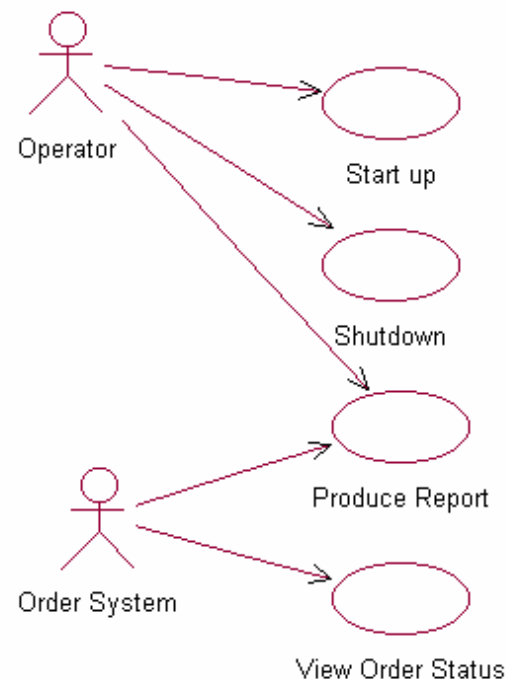
---

- Phần mềm có thể xem xét dưới nhiều góc độ khác nhau
- Không phải hệ thống nào cũng cần tất cả các góc nhìn
- UML và mô hình 4+1 là khung tham chiếu, không phải “quy định cứng nhắc”
- Với dự án nhỏ, nguyên tắc là: “chỉ mô hình hóa vừa đủ”
  - Thường chỉ cần UC view và Logical view



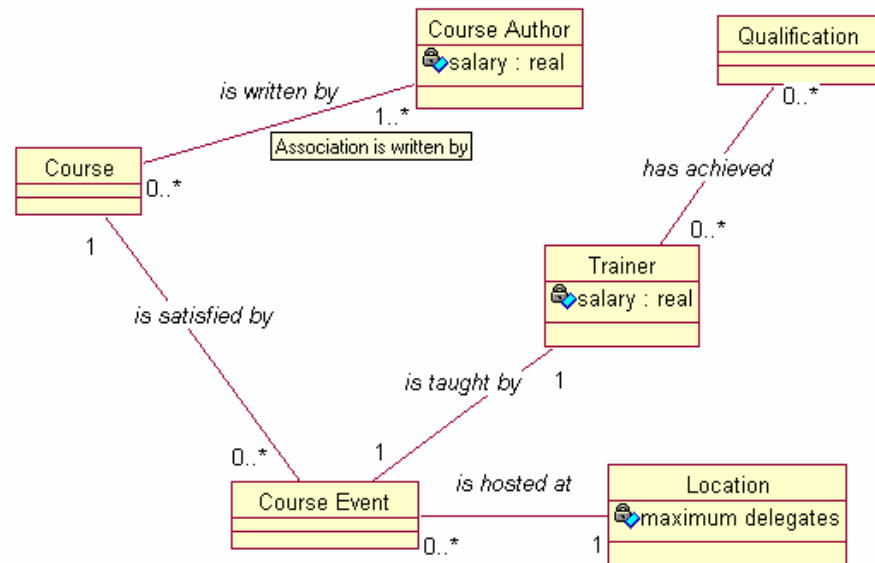
# Sơ đồ ca sử dụng

- **Use case diagram**
- Mô tả hành vi của hệ thống từ góc nhìn của người dùng
- Giúp hiểu các yêu cầu mà hệ thống cần làm được
- Giúp người phát triển (analysts, coders, tests) và khách hàng trao đổi với nhau



# Sơ đồ lớp

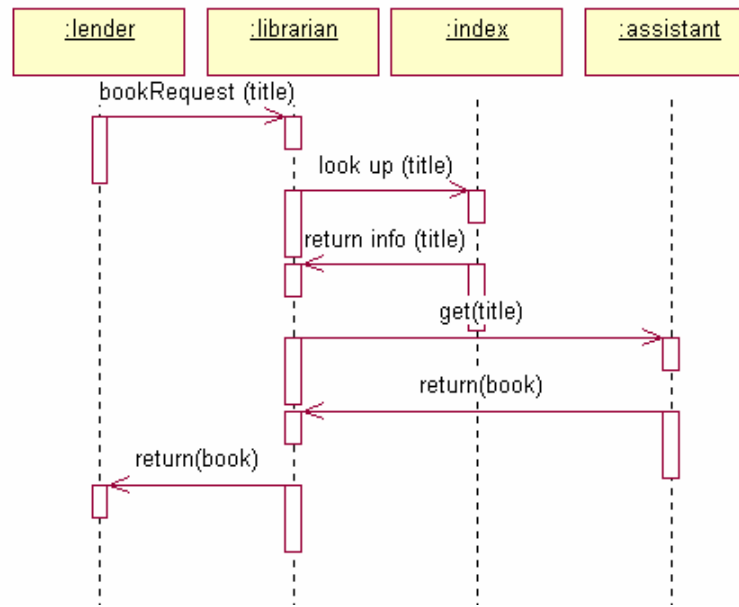
- **Class diagram**
- Sơ đồ quan trọng trong cả phân tích và thiết kế



# Sơ đồ trình tự

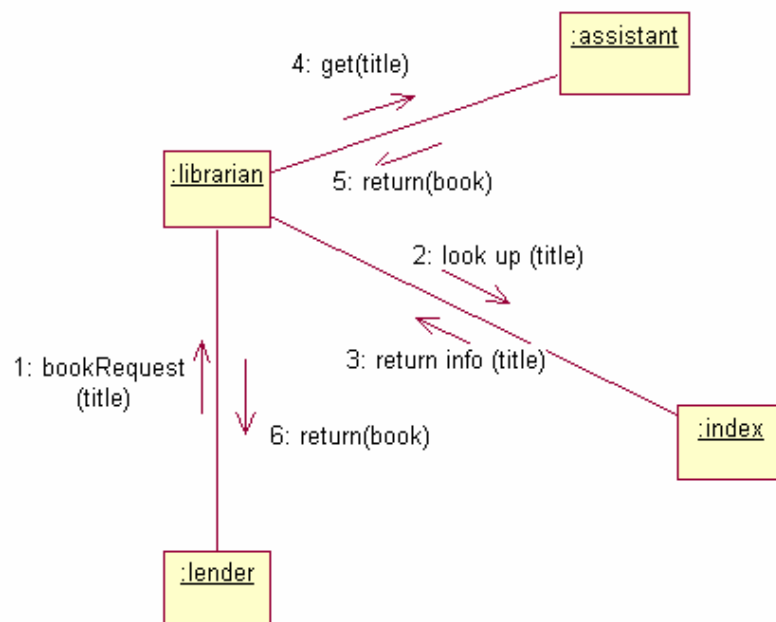
- **Sequence diagram**

- Tác dụng tương tự như sơ đồ giao tiếp, nhưng rõ hơn về thứ tự thực hiện của các thông điệp



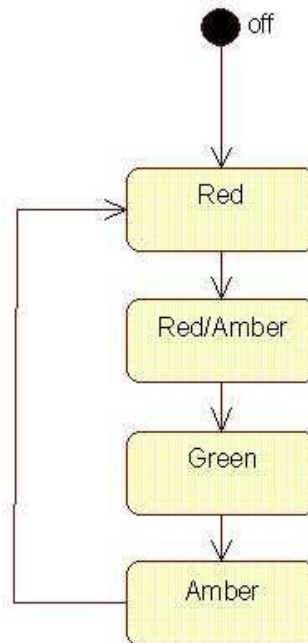
# Sơ đồ giao tiếp

- **Communication diagram** (trong UML 1.0 là Collaboration diagram)
- Mô tả sự tương tác giữa các đối tượng



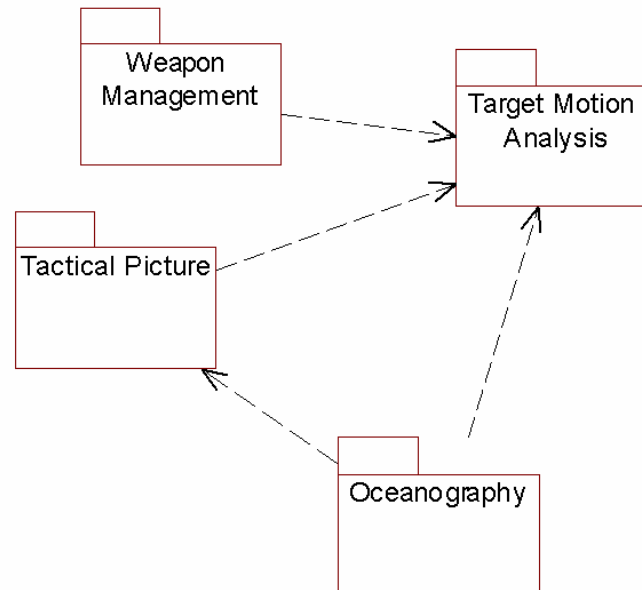
# Sơ đồ máy trạng thái

- **State machine diagram** (trong UML 1.0 là State diagram)
- Một số đối tượng, trong thời điểm nào đó có một trạng thái nhất định



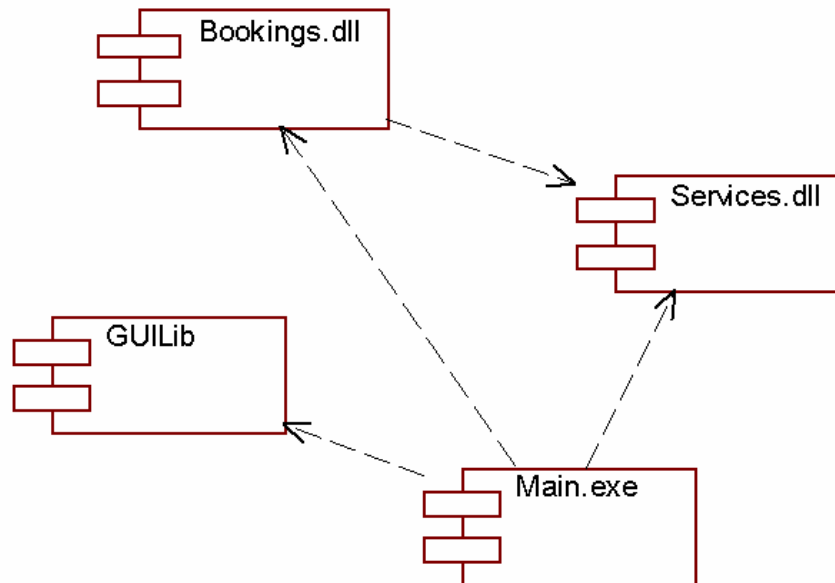
# Sơ đồ gói

- **Package diagram**
- Tương tự như “thư mục” trong quản lý tệp tin
- Có thể có gói các ca sử dụng, gói các lớp



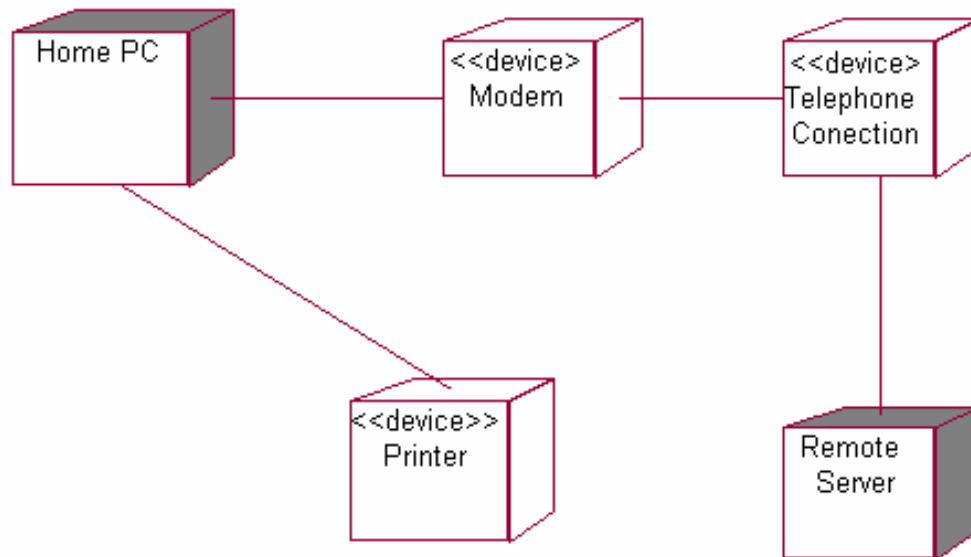
# Sơ đồ thành phần

- **Component diagram**
- Mô tả sự phụ thuộc giữa các thành phần phần mềm (files, libraries, ...)



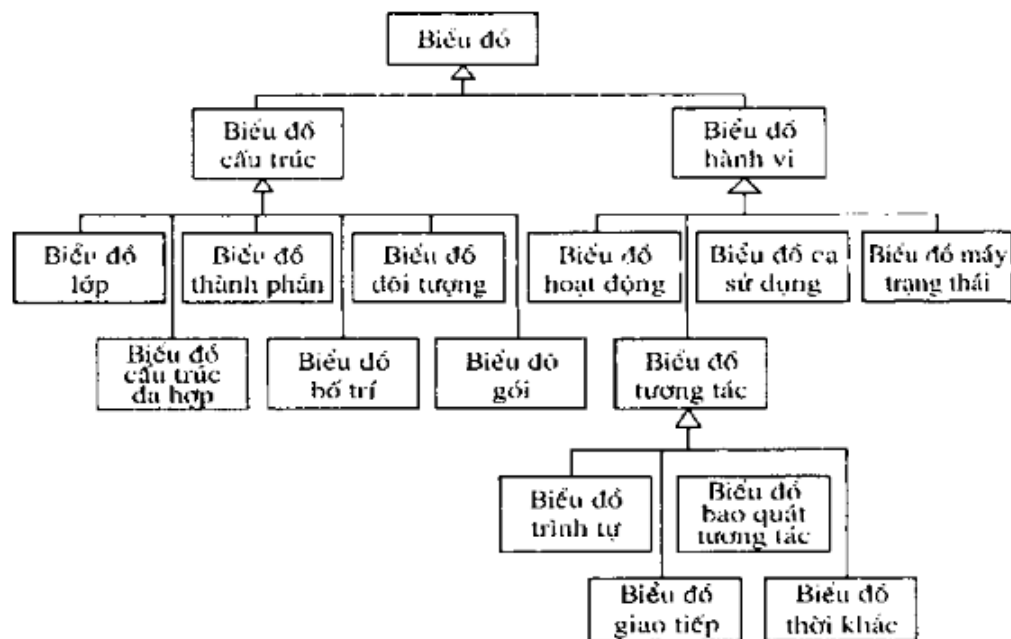
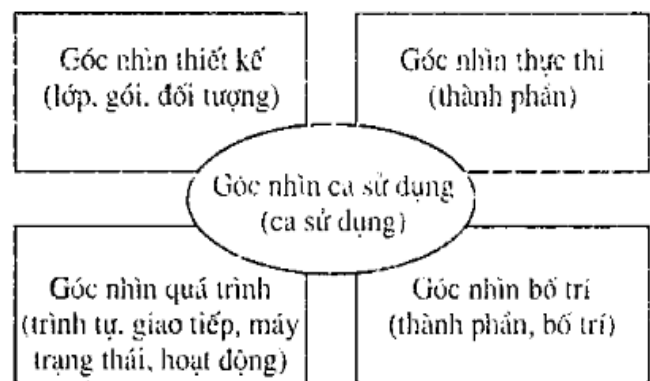
# Sơ đồ triển khai

- **Deployment diagram**
- Mô tả về cách phần mềm của chúng ta sẽ được triển khai





# Tóm tắt UML



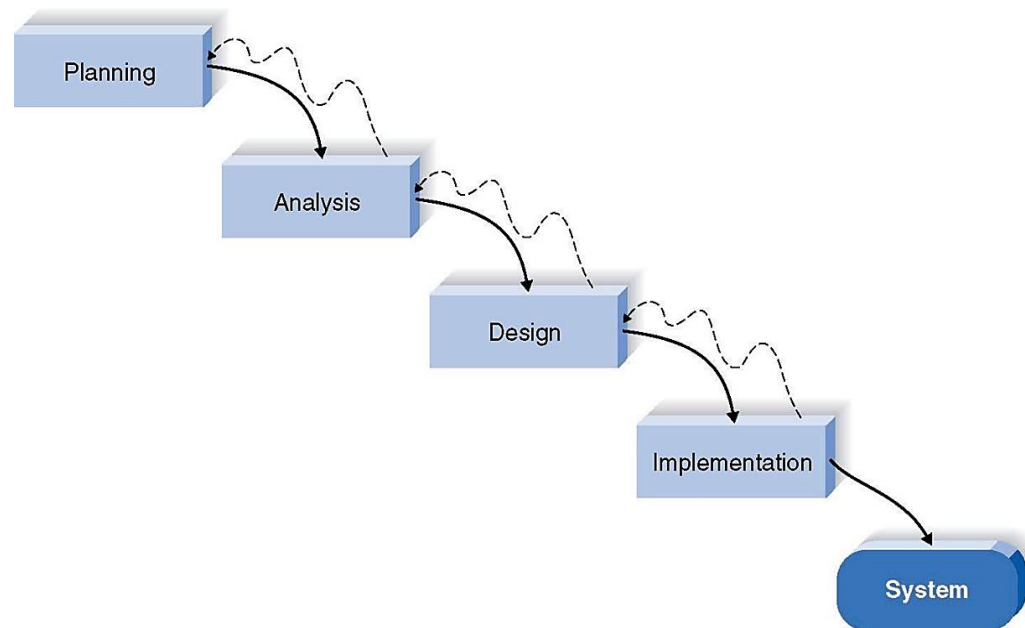
# Phương pháp và quy trình

---

- Phân tích thiết kế hướng đối tượng (OOAD) là phương pháp không phải là một quy trình
- Cần kết hợp nó với một quy trình/ tiến trình cụ thể
- Tiến trình RUP
  - Là một tiến trình sử dụng mô hình hóa hướng đối tượng
  - Quy trình tổng thể gồm nhiều giai đoạn

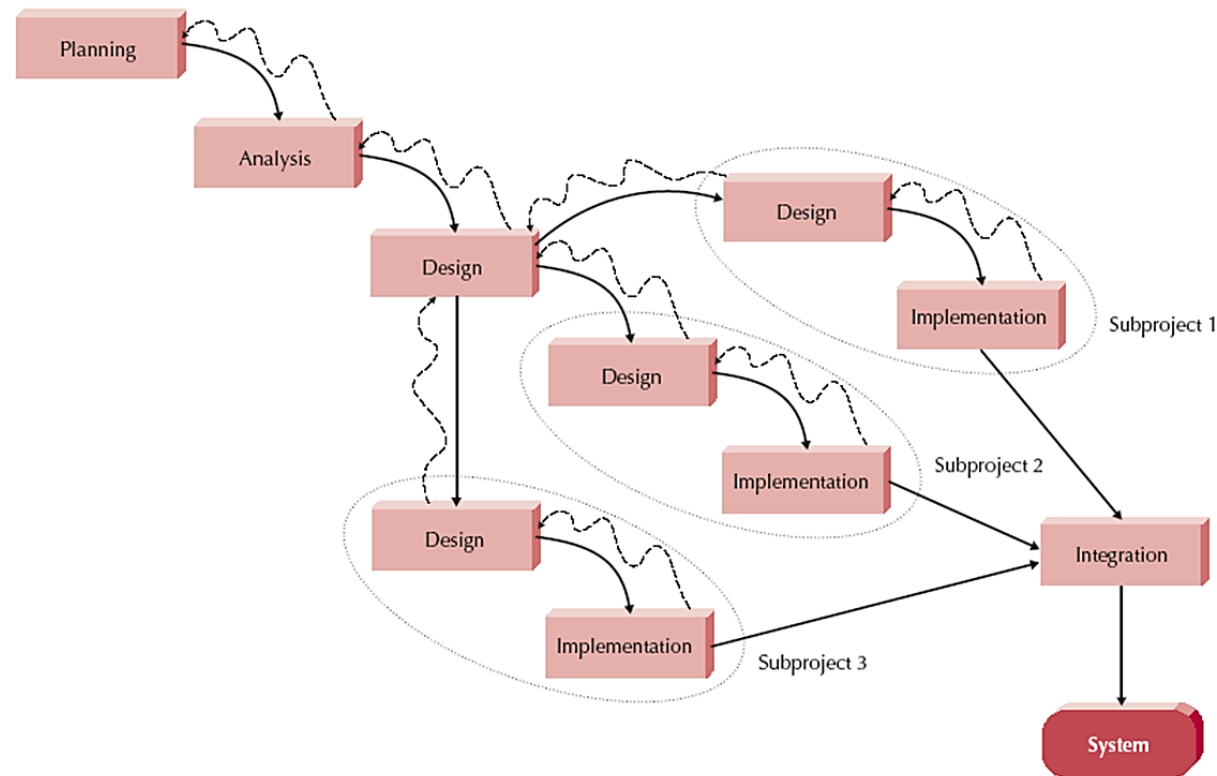
# Thác nước (Waterfall)

- Ưu điểm
  - Các pha được xác định rõ
  - Hỗ trợ tốt cho lên kế hoạch
  - Có kết quả sau mỗi pha
- Nhược điểm
  - Phải hoàn chỉnh toàn bộ các pha trước mới sang pha sau
  - Xử lý lỗi muộn
  - Thời gian chuyển giao lâu



# Phát triển song song

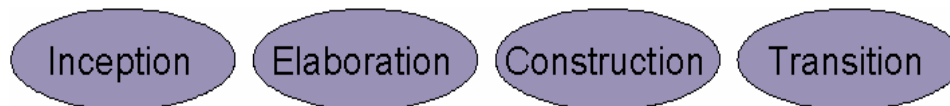
- Ưu điểm
  - Thiết kế tổng thể
  - Giảm thời gian
- Nhược điểm
  - Tích hợp các dự án con không đơn giản



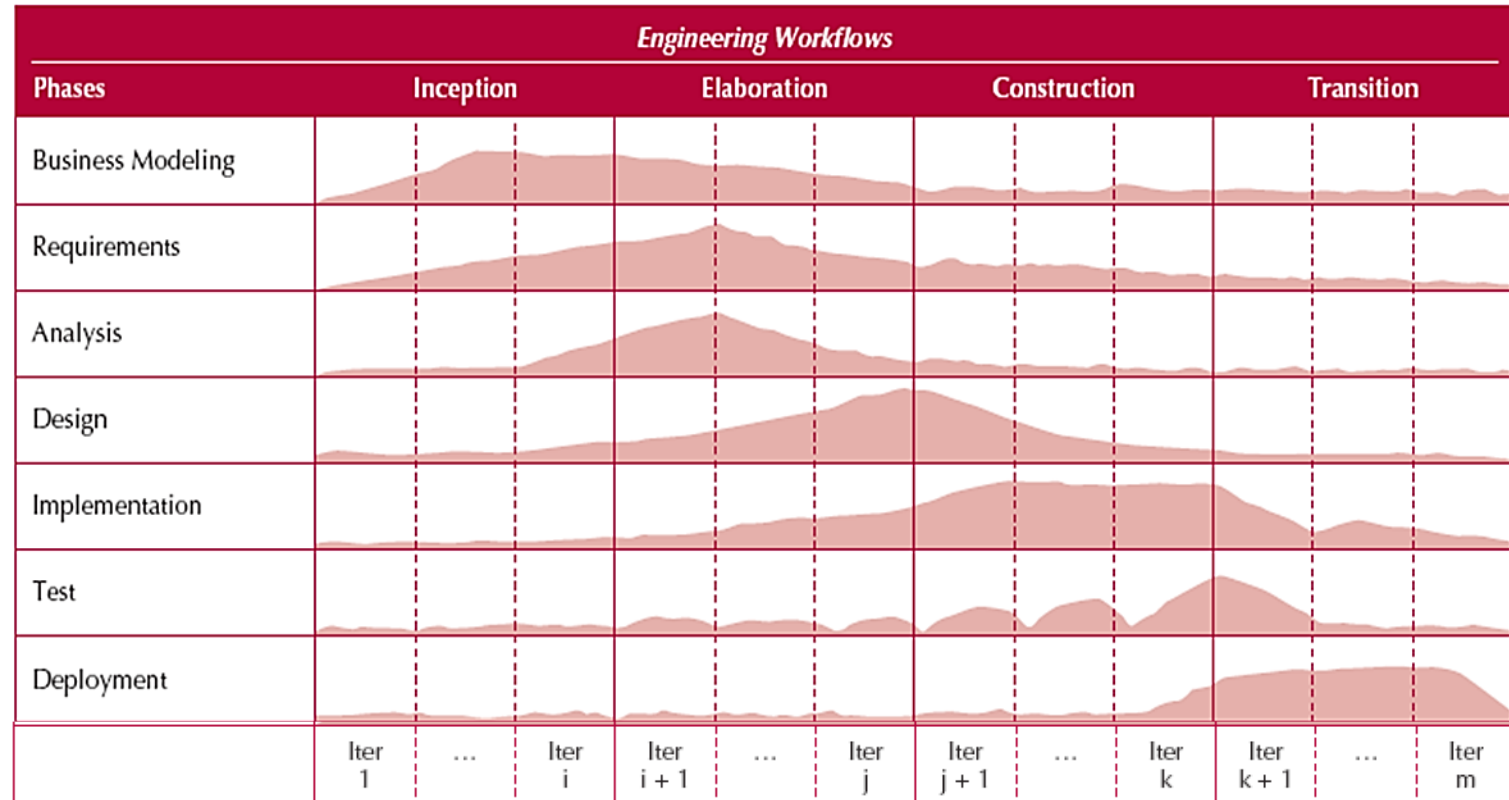
# Tiến trình RUP

---

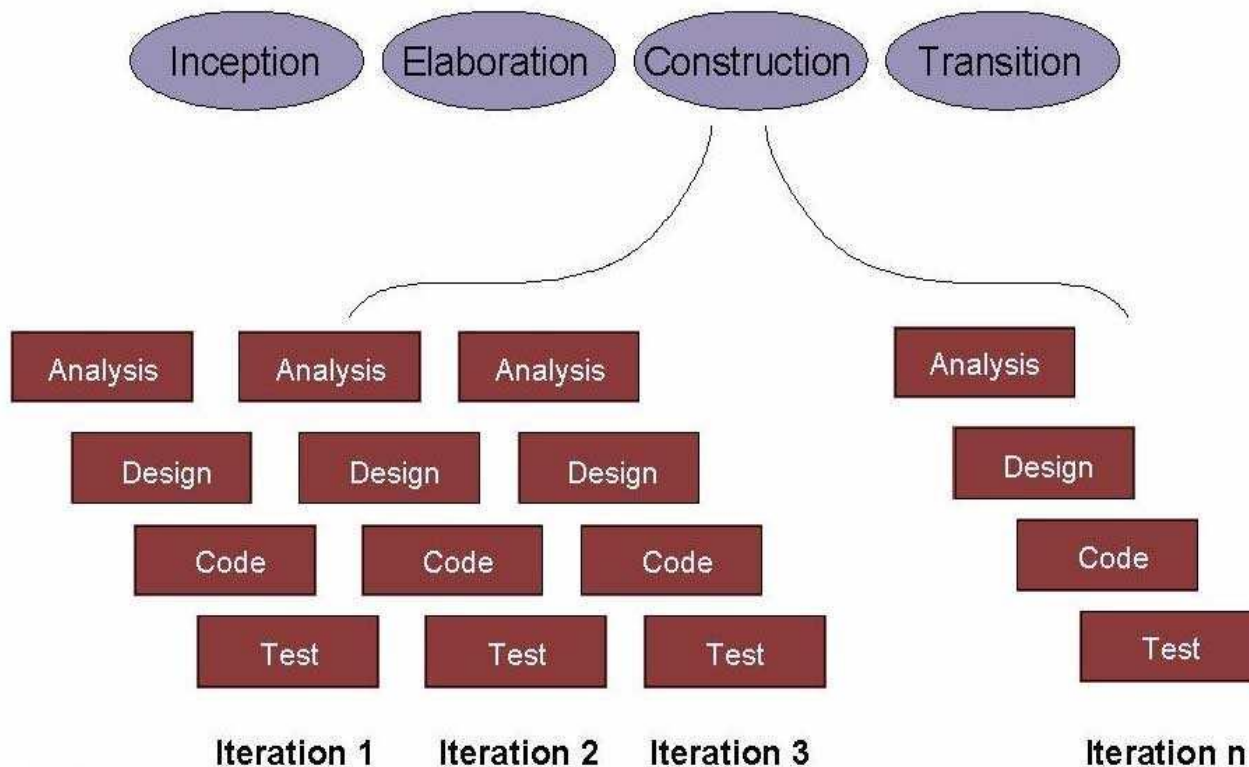
- Thực hiện trong khuôn khổ lặp và tăng trưởng
- Chia thành bốn giai đoạn (phase) chính:
  1. Khởi đầu (Inception)
  2. Triển khai (Elaboration)
  3. Xây dựng (Construction)
  4. Chuyển giao (Transition)



# Tiến trình phát triển RUP



# Ví dụ: Một tiến trình RUP rút gọn



# Ví dụ: Liên hệ RUP và OOAD

Pha	Mục tiêu chính	Công việc cụ thể
<b>Inception</b>	Xác định phạm vi, yêu cầu, tác nhân	<ul style="list-style-type: none"><li>- Viết yêu cầu nghiệp vụ</li><li>- Vẽ Use Case Diagram</li><li>- Xác định chức năng chính</li></ul>
<b>Elaboration</b>	Phân tích lớp, xác định thiết kế logic	<ul style="list-style-type: none"><li>- Phân tích lớp ứng viên</li><li>- Viết CRC</li><li>- Vẽ Class, Sequence Diagrams</li></ul>
<b>Construction</b>	Thiết kế chi tiết và lập trình	<ul style="list-style-type: none"><li>- Thiết kế chi tiết các lớp</li><li>- Mapping từ các lớp sang code</li><li>- Viết các lớp Service, Repo</li></ul>
<b>Transition</b>	Kiểm thử, đóng gói và triển khai	<ul style="list-style-type: none"><li>- Viết hướng dẫn sử dụng</li><li>- Vẽ Deployment Diagram</li><li>- Cấu hình chạy thử</li></ul>



# Tóm tắt

---

1. Giới thiệu môn học
2. Mô hình và mô hình hoá
3. Phân tích thiết kế hướng đối tượng
4. Ngôn ngữ mô hình hoá
5. Tổng quan về UML
6. Phương pháp và quy trình

# Câu hỏi

---

1. Liệt kê 3 lợi ích của mô hình hóa trong phát triển phần mềm?
2. So sánh OOAD với phương pháp thác nước?
3. Hãy nêu sự khác nhau cơ bản giữa Logical View và Physical View trong UML.
4. Trong dự án nhỏ (ví dụ Quản lý thư viện), sơ đồ UML nào là quan trọng nhất và không thể thiếu?
5. Giả sử bạn phải thiết kế hệ thống Quản lý thư viện, khi nào thì góc nhìn Process trở nên quan trọng?