

PHÂN TÍCH VÀ THIẾT KẾ HƯỚNG ĐỐI TƯỢNG

Th.S. Võ Đức Quang

Bộ môn KHMT&CNPM, Viện Kỹ thuật và công nghệ

✉ quangvd.cntt.dhv@gmail.com, quangvd@vinhuni.edu.vn

☎ 0989.891.418

Chương 2. Phát triển phần mềm (6t)

- 2.1. Tổng quan quản lý dự án phần mềm
- 2.2. Một số quy trình phát triển phần mềm thông dụng
 - Mô hình thác nước (Waterfall model)
 - Mô hình nguyên mẫu (Prototyping model)
 - Mô hình xoắn ốc (Spiral model)
 - Mô hình nhanh gọn (Agile model)
 - Mô hình hợp nhất (Unified model)
- 2.3. Tiếp cận hệ thống hướng chức năng
- 2.4. Tiếp cận hệ thống hướng đối tượng

2.1. Tổng quan quản lý dự án phần mềm

- **Dự án phần mềm (Project)** là tập hợp các công việc được thực hiện bởi một tập thể nhằm đạt được một **kết quả** như dự kiến, trong **thời gian** dự kiến, với một **kinh phí** dự kiến
- Người quản lý dự án (PM-Project Managers): theo dõi, giám sát tất cả các công việc và vai trò, cái mà cần được điều phối
- **Làm thế nào để dự án THÀNH CÔNG?**



2.1. Tổng quan quản lý dự án phần mềm

- Một số nguyên nhân dự án thất bại
 - Dự án không có tính thực tế và không khớp
 - Ước tính không chính xác nguồn lực cần thiết cho dự án
 - Xác định yêu cầu hệ thống không đúng
 - Báo cáo tình trạng dự án sơ sài
 - Không quản lý độ rủi ro
 - Việc giao tiếp khách hàng, người sử dụng và người phát triển dự án không tốt
 - Sử dụng công nghệ chưa phát triển
 - Không có khả năng xử lý độ phức tạp của dự án
 - Phát triển thực hành không có hệ thống
 - Thiếu kinh nghiệm trong việc quản lý dự án
 - Các bên liên quan mang tính chính trị
 - Các áp lực mang tính thương mại

2.1. Tổng quan quản lý dự án phần mềm

- Quản lý dự án phần mềm là các hoạt động trong **lập kế hoạch**, **giám sát** và **điều khiển** tài nguyên dự án, thời gian thực hiện, các rủi ro và quy trình thực hiện dự án nhằm đảm bảo thành công cho dự án.



2.1. Tổng quan quản lý dự án phần mềm

- Quản lý dự án phần mềm cần đảm bảo cân bằng giữa ba yếu tố: **thời gian**, **tài nguyên** và **chất lượng**. → gọi là tam giác dự án.
 - Tài nguyên: con người, kinh phí, công nghệ,...
 - Thời gian: bắt đầu, kết thúc, giai đoạn
 - Chất lượng: đáp ứng yêu cầu? giá trị hữu hình, vô hình?
- Quy trình dự án: vận dụng những kiến thức, kỹ năng và kỹ thuật công nghệ vào hoạt động của dự án để đạt được mục tiêu của dự án đặt ra (đã được tiêu chuẩn hóa)

2.1. Tổng quan quản lý dự án phần mềm

- Một số hoạt động trong quản lý dự án
 - Xác định dự án:
 - Bởi người dùng nghiệp vụ hoặc IT-er hoặc cả hai
 - Giá trị nghiệp vụ:
 - Hữu hình: tiết kiệm 2% chi phí vận hành,...
 - Vô hình: nâng cao chất lượng dịch vụ,...
 - Yêu cầu hệ thống
 - Tài liệu mô tả lý do và giá trị mà việc xây dựng hệ thống mới đem lại
 - Ảnh hưởng bởi 5 yếu tố: nhà đầu tư dự án, nhu cầu nghiệp vụ, yêu cầu nghiệp vụ, giá trị nghiệp vụ, vấn đề đặc biệt
 - Phân tích tính khả thi
 - Rủi ro? Khắc phục rủi ro?
 - Tính khả thi
 - kỹ thuật: Can we build it?
 - kinh tế: Should we build it?
 - tổ chức sử dụng: Will they use it?

2.1. Tổng quan quản lý dự án phần mềm

- Tính khả thi về kỹ thuật
 - Xác định rủi ro về
 - Chức năng: người phân tích có quen thuộc với nghiệp vụ không?
 - Công nghệ:
 - Quy mô dự án: dự án lớn thì rủi ro càng cao
 - Khả năng tương thích: vấn đề tích hợp
- Tính khả thi về kinh tế
 - Xác định các chi phí và lợi ích
 - Định lượng chi phí và lợi ích
 - Xác định dòng tiền
 - Xác định giá trị: *giá trị hiện tại, lợi tức đầu tư, điểm hòa vốn*

2.1. Tổng quan quản lý dự án phần mềm

- Quản lý phạm vi của dự án
 - Thêm các yêu cầu nghiệp vụ mới
 - Nhân lực, kinh phí, tiến độ
 - ✓ Xác định các yêu cầu ngay từ đầu
 - ✓ Cân nhắc, tính toán chỉ thay đổi nhu cầu cần thiết
 - ✓ Trì hoãn một số thay đổi để cải tiến trong tương lai
 - ✓ Kiểm soát khung thời gian
- Tạo và quản lý kế hoạch làm việc
- Nhân sự dự án
 - Số lượng, kỹ năng, ...
 - Điều lệ dự án
 - Quản lý nhóm, mục tiêu công việc
 - Giảm thiểu xung đột
 - Tạo động lực làm việc: lương, thưởng, ghi nhận, môi trường làm việc, thưởng phạt công bằng, trao quyền tự chủ và tin tưởng

2.1. Tổng quan quản lý dự án phần mềm

- Công cụ hỗ trợ dự án
 - Quản lý công việc
 - Quản lý thời gian
- Biểu đồ Gantt

2.2. Một số quy trình phát triển phần mềm

- Quy trình phát triển phần mềm là gì?
→ Một tập có cấu trúc (có trật tự) các hoạt động cần thiết để phát triển một hệ thống phần mềm
- Không tồn tại một quy trình PTPM lý tưởng duy nhất phù hợp cho mọi bài toán, yêu cầu thực tế
- → Lựa chọn quy trình phù hợp?

infogila.com



2.2. Một số quy trình phát triển phần mềm

- Lựa chọn quy trình PTPM
 - Kiểu của hệ thống phần mềm cần được xây dựng
 - Xây dựng mới từ đầu >< Nâng cấp, chỉnh sửa hệ thống có sẵn
 - Kiểu thông thường, phổ biến >< Kiểu tùy biến, đặc thù
 - Các yêu cầu phần mềm xác định >< Các yêu cầu phần mềm thay đổi (nhANH chóng)
 - Hệ thống trọng yếu (critical) >< Hệ thống nghiệp vụ, kinh doanh
 - Quy mô của dự án PTPM, Quy mô (nguồn lực) của nhóm PTPM, Thời gian thực hiện dự án PTPM
 - Các đặc điểm của nhóm PTPM
 - Kinh nghiệm, Động cơ (+ sự khuyến khích), Thái độ làm việc (nỗ lực)
 - Kinh phí thực hiện dự án PTPM

2.2. Một số quy trình phát triển phần mềm

- Các hoạt động cơ bản của quy trình PTPM
 - Phân tích tính khả thi (Feasibility study)
 - Phân tích và đặc tả yêu cầu (Requirements analysis)
 - Thiết kế (Design)
 - Thực hiện, cài đặt (Implementation)
 - Kiểm thử (Testing)
 - Triển khai (Deployment)
 - Bảo trì (Maintenance)

2.2. Một số quy trình phát triển phần mềm

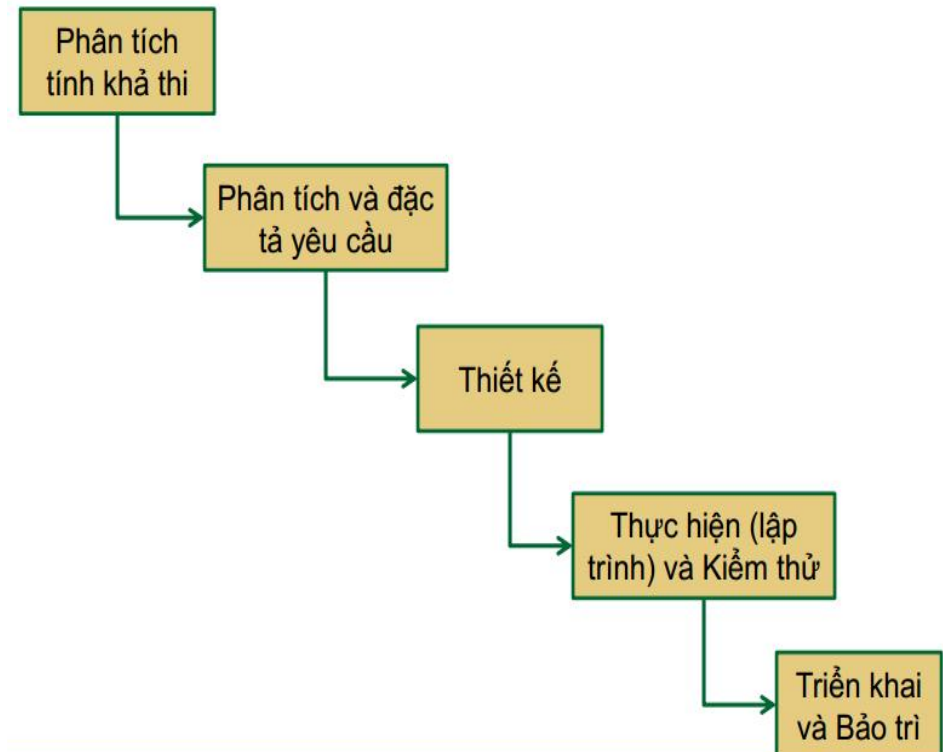
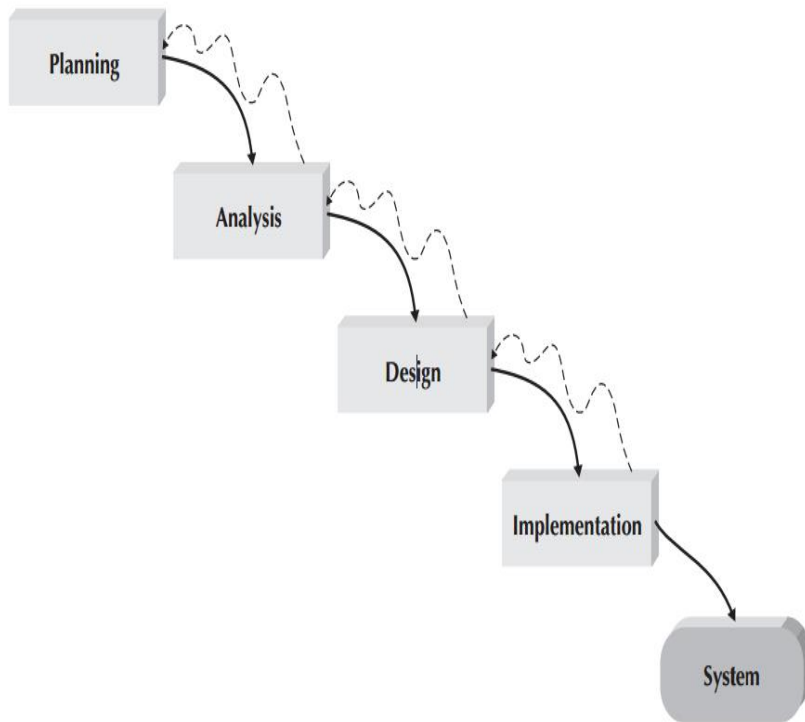
- Phân loại
 - Hướng chức năng (Process oriented)
 - Hướng dữ liệu làm trung tâm (Data centered)
 - Hướng cấu trúc (Structured)
 - Mô hình thác nước (water fall)
 - Mô hình song song (Parallel)
 - Hướng đối tượng (Object oriented)
 - Phát triển ứng dụng nhanh (Rapid action development)
 - Theo giai đoạn (Phased)
 - Theo nguyên mẫu (Prototyping)
 - Nguyên mẫu vứt bỏ (Throwaway Prototyping)
 - Phát triển linh hoạt (Agile Development)
 - Extreme Programming
 - SCRUM

2.2. Một số quy trình phát triển phần mềm

- Mô hình thác nước (Waterfall model)
- Mô hình nguyên mẫu (Prototyping model)
- Mô hình xoắn ốc (Spiral model)
- Mô hình nhanh gọn (Agile model)
- Mô hình hợp nhất (Unified model)

2.2. Một số quy trình phát triển phần mềm

- Water fall model



2.2. Một số quy trình phát triển phần mềm

- Water fall model

- 1970 Winston Royce (hiện nay vẫn được dùng phổ biến)
- Các giai đoạn (phases) không giao nhau, không lặp lại

- Ưu:

- đơn giản, dễ hiểu, dễ áp dụng
- Các tài liệu được hoàn thành sau mỗi giai đoạn
- PM thuận lợi trong lập kế hoạch và kiểm soát dự án

- Nhược:

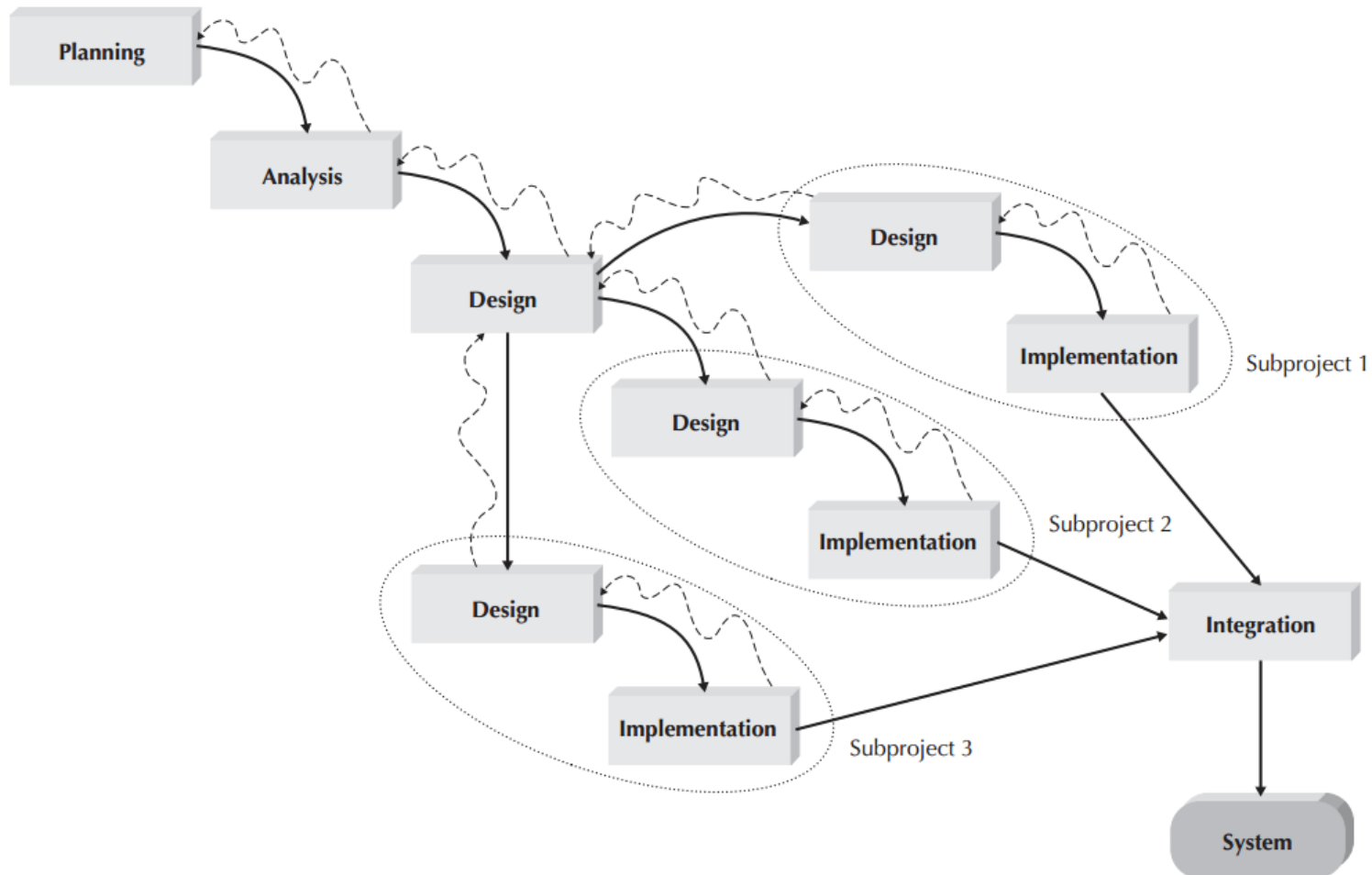
- Chỉ phù hợp đối với các bài toán thực tế khi mà các yêu cầu phần mềm được xác định rõ ràng, đầy đủ và cố định từ đầu
- Không phù hợp đối với các dự án kéo dài và tiếp diễn lâu
- Có thể có nhiều nguy cơ (risk) và không chắc chắn (uncertainty)
- Khó (không thể) sớm có các kết quả (phiên bản) ban đầu của phần mềm

2.2. Một số quy trình phát triển phần mềm

- Water fall model
 - Nên sử dụng khi:
 - **Các yêu cầu phần mềm được xác định rõ ràng, đầy đủ và cố định**
 - Định nghĩa về sản phẩm (hệ thống phần mềm) không thay đổi
 - Các công nghệ liên quan cần thiết được nắm vững
 - Các nguồn lực và kinh nghiệm của nhóm PTPM đủ đáp ứng
 - Thời gian thực hiện dự án ngắn (không kéo dài)

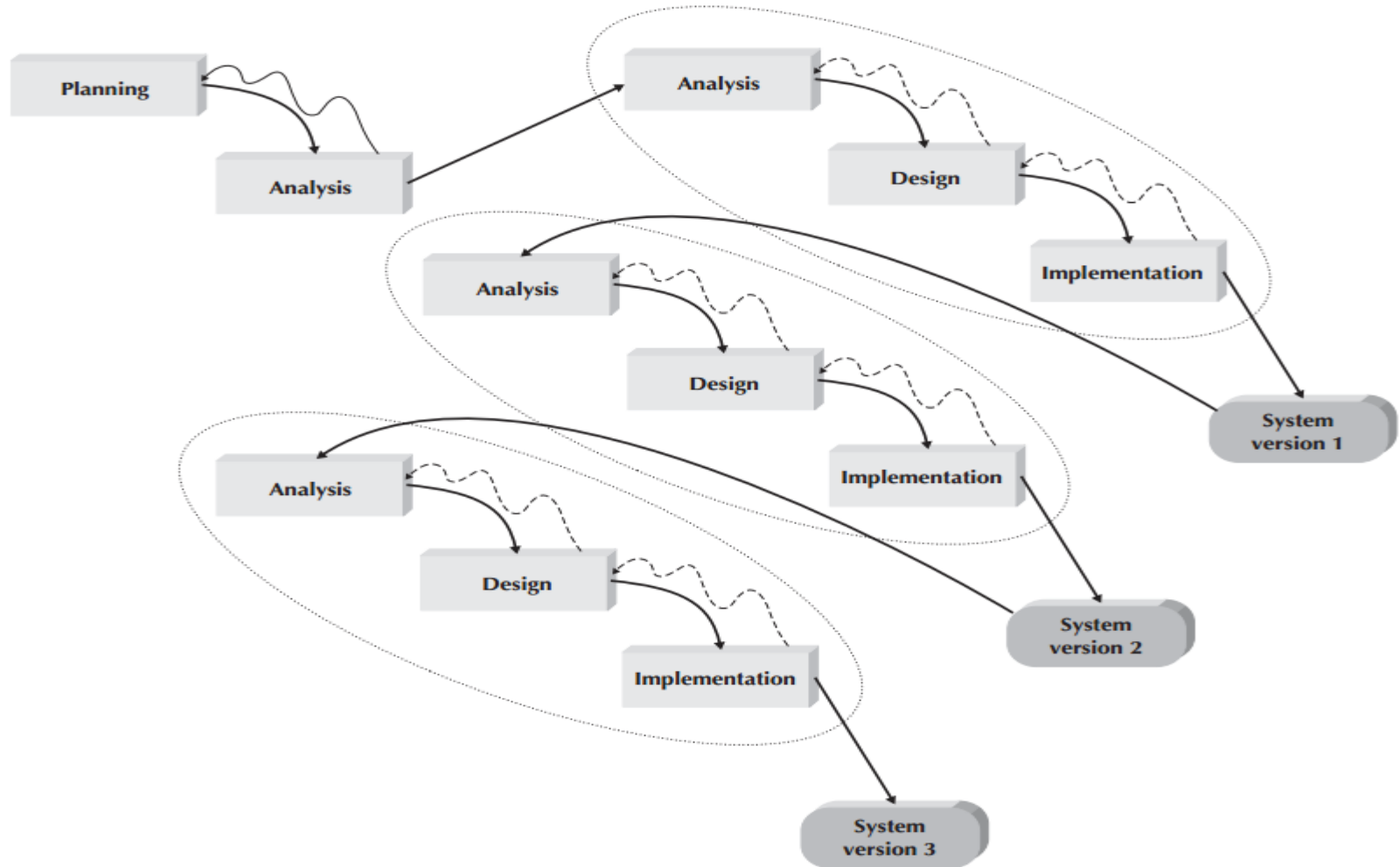
2.2. Một số quy trình phát triển phần mềm

- Parallel model



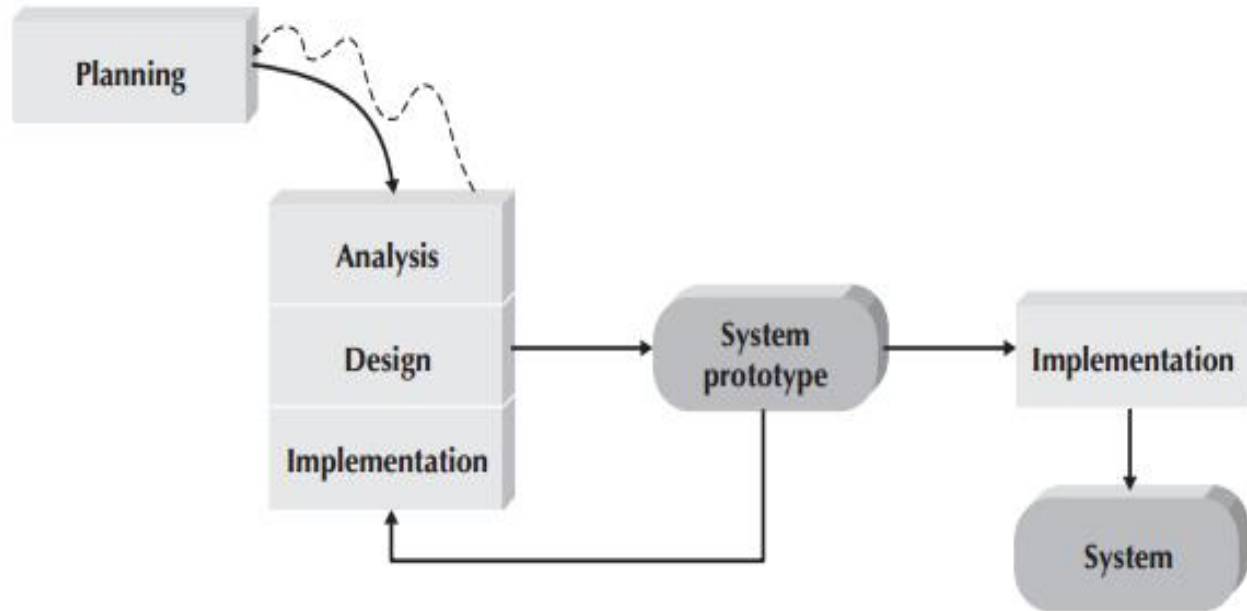
2.2. Một số quy trình phát triển phần mềm

- Phased Development model



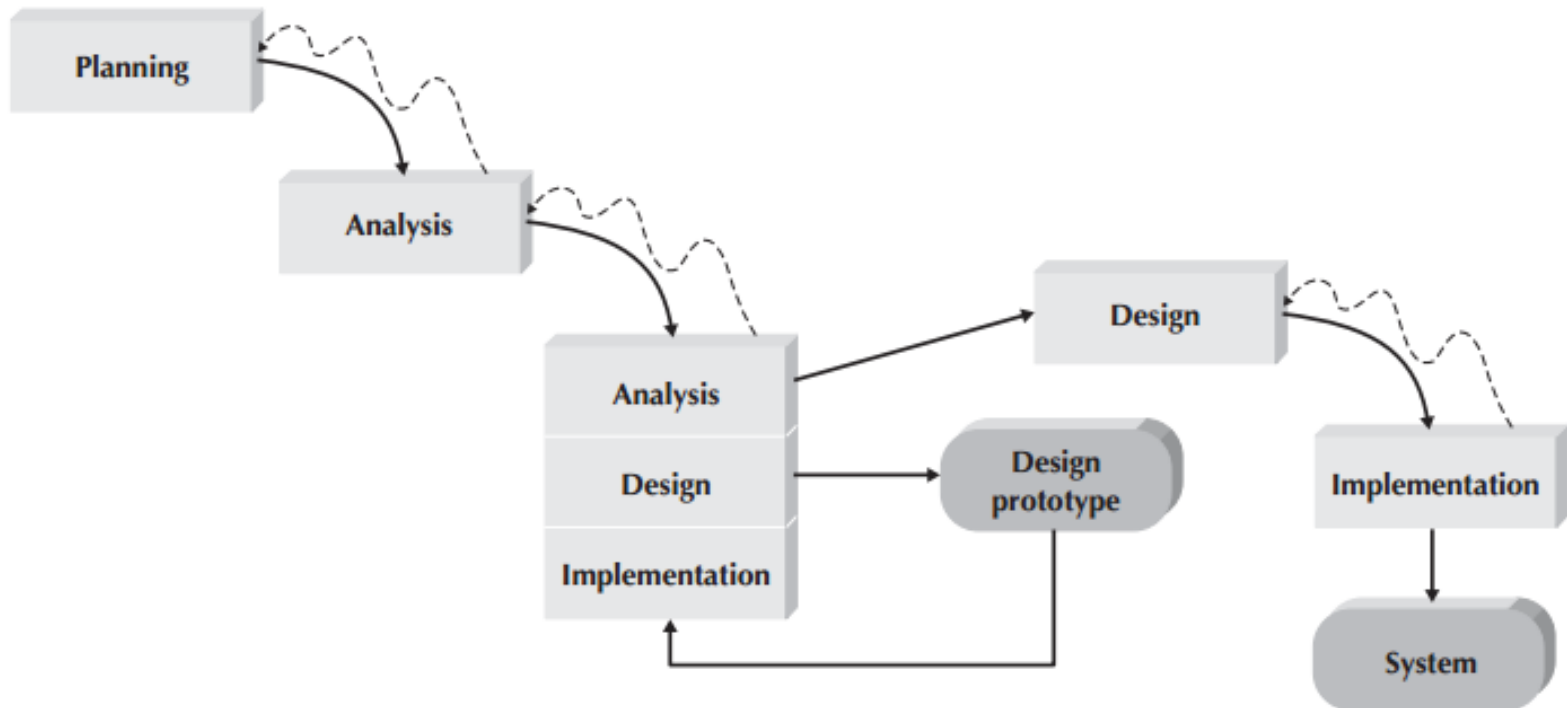
2.2. Một số quy trình phát triển phần mềm

- Prototyping model



2.2. Một số quy trình phát triển phần mềm

- Throwaway Prototyping model



2.2. Một số quy trình phát triển phần mềm

■ Prototyping model

- Xây dựng một (hoặc một số các) nguyên mẫu (prototype) để hiểu chính xác các yêu cầu phần mềm
- Nguyên mẫu (prototype) mới thu được từ việc đánh giá các nguyên mẫu trước đó
- Giúp khách hàng có được “cảm nhận thực tế” về hệ thống phần mềm, hiểu chính xác hơn về các yêu cầu của hệ thống phần mềm mong muốn
- Phù hợp với việc phát triển các hệ thống phần mềm lớn và phức tạp (khi không có quy trình thu thập yêu cầu hoặc hệ thống sẵn có nào giúp xác định các yêu cầu phần mềm)
- Một nguyên mẫu thường không phải là một hệ thống phần mềm hoàn chỉnh/hoàn thiện, và rất nhiều các chi tiết không được xây dựng trong nguyên mẫu

2.2. Một số quy trình phát triển phần mềm

- Prototyping model

- Ưu điểm

- Người sử dụng được tham gia tích cực vào trong quá trình PTPM
 - Người sử dụng hiểu rõ hơn về hệ thống đang được xây dựng
 - Các lỗi, vấn đề có thể được phát hiện từ (rất) sớm
 - Sớm có được các phản hồi đánh giá từ người sử dụng, giúp có được các giải pháp PTPM tốt hơn
 - Các chức năng còn thiếu, không rõ ràng, khó thao tác có thể được phát hiện sớm

- Nhược điểm

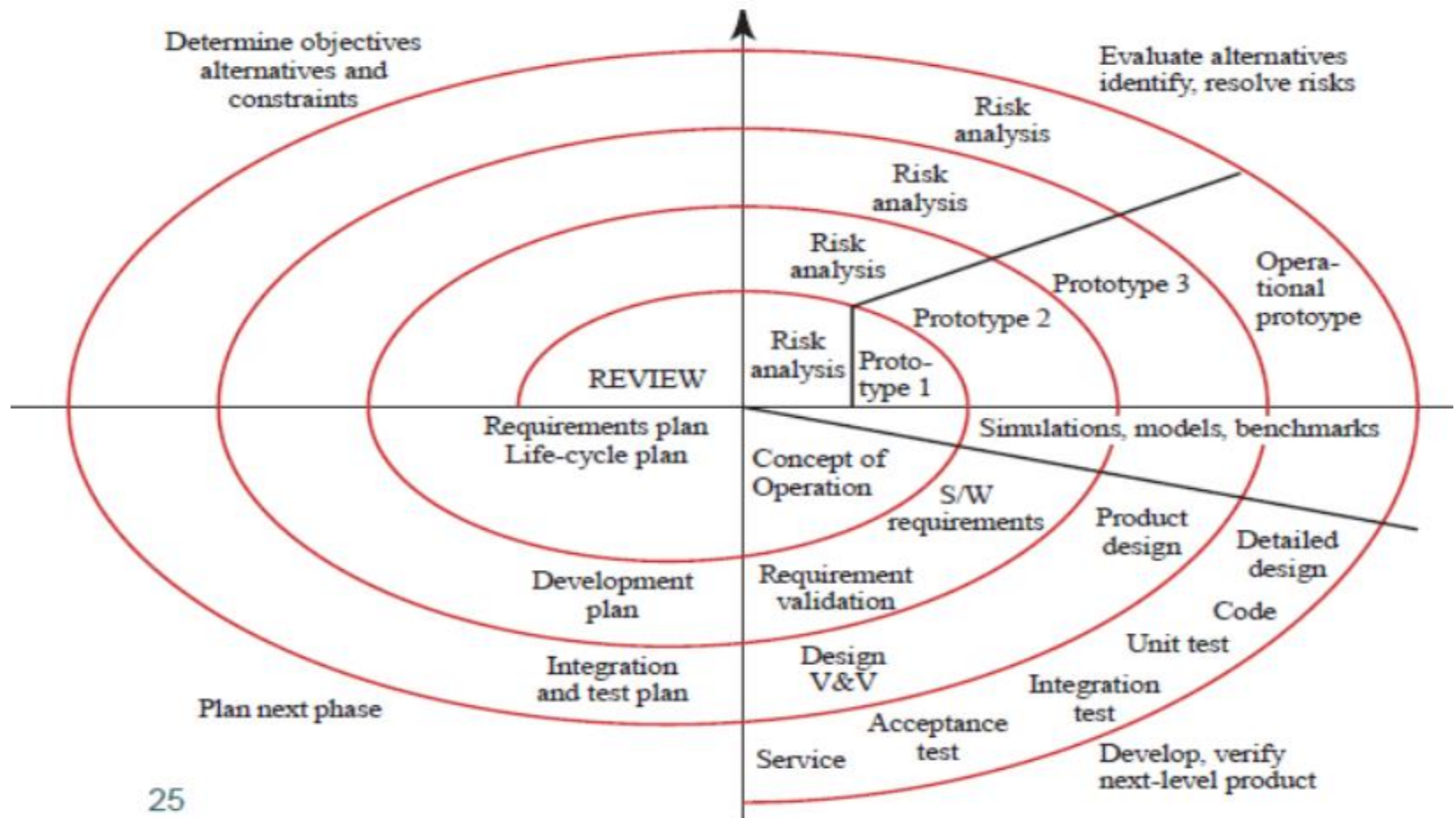
- Người sử dụng có thể nghĩ rằng việc phát triển phần mềm là dễ dàng, và vì vậy trở nên không nhất quán trong việc diễn đạt các yêu cầu
 - Không có việc lập kế hoạch ngay từ đầu, có thể dẫn đến các vấn đề về quản lý dự án: không xác định được thời hạn hoàn thành, ngân sách và các kết quả bàn giao
 - Mô hình này thường dẫn đến kéo dài quá trình PTPM
 - Các người phát triển có xu hướng bàn giao một nguyên mẫu hoạt động cơ bản, thay vì bàn giao một sản phẩm hoàn thiện thực sự

2.2. Một số quy trình phát triển phần mềm

- Dùng Prototyping model khi:
 - Các yêu cầu phần mềm không thể được xác định tại thời điểm bắt đầu dự án
 - **Những người sử dụng (vì các lý do khác nhau) không thể diễn đạt các yêu cầu của họ một cách rõ ràng**
 - Mô hình PTPM này rất phù hợp để phát triển “cảm nhận” (look and feel) hoặc giao diện sử dụng của hệ thống, bởi vì các đặc điểm này rất khó để miêu tả bằng tài liệu, mà thường thông qua việc dùng trải nghiệm
 - Khách hàng yêu cầu chứng minh tính khả thi của hệ thống
 - Cần có các demos cho các cấp quản lý ở mức cao
 - Các vấn đề về công nghệ cần được thử nghiệm, kiểm tra

2.2. Một số quy trình phát triển phần mềm

- Spiral model



2.2. Một số quy trình phát triển phần mềm

- Spiral model

- Một mô hình phát triển **tiến hóa**, lai ghép của đặc điểm phát triển **lặp** (iterative) của **Prototyping model** và phát triển theo các bước **tuần tự** (sequential) **Waterfall model**
 - **Có chú trọng vào việc phân tích nguy cơ (risk analysis)**
- Ưu điểm:
 - Chú trọng phân tích rủi ro → giảm thiểu rủi ro dự án PTPM
 - Phù hợp đối với các dự án lớn và quan trọng đặc biệt
 - Các chức năng mới có thể được bổ sung vào sau
 - Các phiên bản đầu của hệ thống phần mềm được tạo ra sớm
- Nhược điểm:
 - Chi phí để áp dụng cao (thời gian, nguồn lực, tiền bạc)
 - Việc phân tích rủi ro (risk analysis) đòi hỏi kỹ năng và kinh nghiệm cao
 - Thành công của dự án phụ thuộc nhiều vào giai đoạn phân tích rủi ro
 - Không phù hợp cho các dự án nhỏ

2.2. Một số quy trình phát triển phần mềm

- Dùng Spiral model khi nào?
 - **Khi việc đánh giá (phân tích) các chi phí và các rủi ro là quan trọng**
 - Đối với các dự án có độ rủi ro trung bình đến cao
 - Các người sử dụng không chắc chắn về các nhu cầu của họ
 - Các yêu cầu phần mềm phức tạp và lớn
 - Cần phát triển một dòng sản phẩm mới (New product line)
 - Mong muốn có các thay đổi quan trọng (cần nghiên cứu và khảo sát cẩn thận)

2.2. Một số quy trình phát triển phần mềm

- Agile model



- Extreme Programming
- SCRUM

2.2. Một số quy trình phát triển phần mềm

- Agile model

- Kiểu mô hình tăng cường (incremental) và lặp lại (iterative) qua các vòng phát triển nhanh
- Phiên bản tăng cường được cải tiến tính năng từ phiên bản trước đó
- **Được sử dụng cho các dự án PTPM đòi hỏi thời gian hoàn thành nhanh chóng**
 - (Extreme Programming – XP) là một phương pháp phát triển phần mềm nổi tiếng thuộc nhóm mô hình nhanh lẹ

- Ưu điểm

- Trao đổi thường xuyên giữa nhóm phân tích nghiệp vụ và nhóm lập trình
- **Thích nghi (đáp ứng) nhanh với các yêu cầu thay đổi**

- Nhược điểm

- Đối với dự án lớn khó đánh giá được các chi phí (effort) cần thiết tại thời điểm bắt đầu tiến trình PTPM
- Ít chú trọng đến các thiết kế và tài liệu cần thiết
- Chỉ các người lập trình có kinh nghiệm (senior programmers) mới có khả năng đưa ra các quyết định cần thiết trong quá trình phát triển

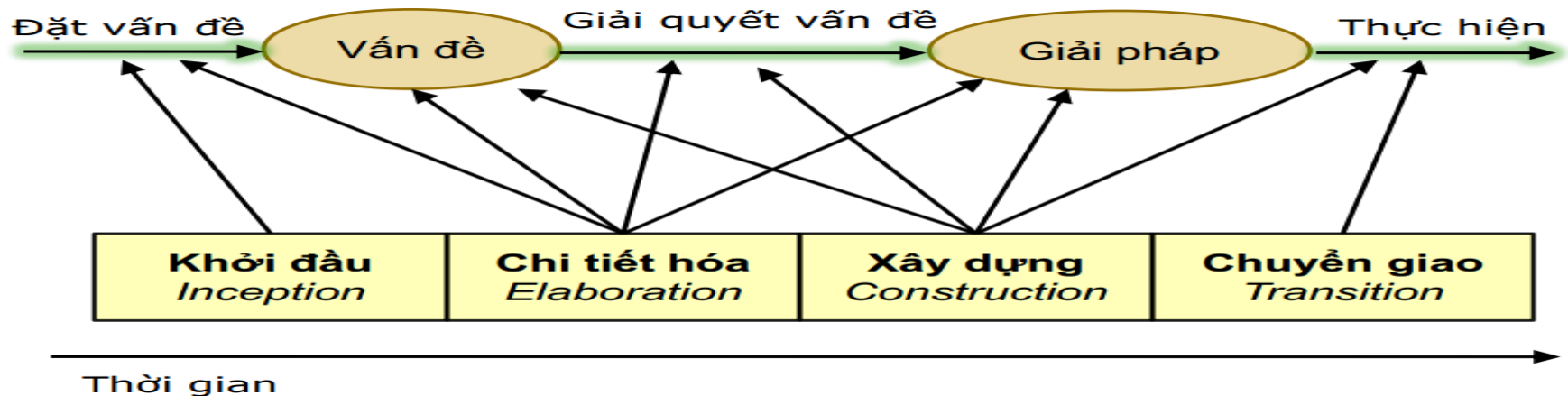
2.2. Một số quy trình phát triển phần mềm

- Dùng Agile model khi nào?
 - Khi các thay đổi cần được bổ sung và thực hiện với chi phí thấp, nhờ việc thường xuyên tạo ra các phiên bản tăng cường
 - Để thực hiện một tính năng mới (a new feature), những người lập trình chỉ cần tốn thời gian vài ngày, hoặc thậm chí là chỉ vài giờ để thực hiện
 - Mô hình nhanh lẹ cho phép các nhu cầu của người dùng sẽ có thể (thường xuyên) thay đổi. Các tính năng luôn luôn có thể được bổ sung hoặc loại bỏ dựa trên các phản hồi, giúp đem lại cho khách hàng sản phẩm mong muốn
 - Cả người phát triển và người sử dụng hệ thống đều cảm thấy họ được tự do về thời gian và các lựa chọn

2.2. Một số quy trình phát triển phần mềm

- Mô hình hợp nhất (Unified model)

- Góc nhìn quản lý dự án
- Góc nhìn kỹ thuật



- UML: The Unified Modeling Language
- The Unified Process
- Quy trình RUP (Rational Unified Process) là một quy trình mô hình hóa với UML

2.2. Một số quy trình phát triển phần mềm

- Đánh giá lựa chọn quy trình phù hợp

Ability to Develop Systems	Structured Methodologies		RAD Methodologies			Agile Methodologies	
	Waterfall	Parallel	Phased	Prototyping	Throwaway Prototyping	XP	SCRUM
With Unclear User Requirements	Poor	Poor	Good	Excellent	Excellent	Excellent	Excellent
With Unfamiliar Technology	Poor	Poor	Good	Poor	Excellent	Good	Good
That Are Complex	Good	Good	Good	Poor	Excellent	Good	Good
That Are Reliable	Good	Good	Good	Poor	Excellent	Excellent	Excellent
With a Short Time Schedule	Poor	Good	Excellent	Excellent	Good	Excellent	Excellent
With Schedule Visibility	Poor	Poor	Excellent	Excellent	Good	Excellent	Excellent

Chương 2. Phát triển phần mềm

- 2.3. Tiếp cận hệ thống hướng chức năng
 - Phân rã chức năng
 - Thực thể-Chức năng
 - Luồng dữ liệu
 - ...
- 2.4. Tiếp cận hệ thống hướng đối tượng
 - Các khái niệm cơ bản: Lớp, đối tượng, phương thức, thông điệp, kế thừa, đa hình
 - PTTK hệ thống theo hướng đối tượng

2.4. Tiếp cận hệ thống hướng chức năng



- Hệ thống thông tin = Dữ liệu + Xử lý

Tiếp cận hệ thống



Tiếp cận Phương pháp phân tích thiết kế HTTT

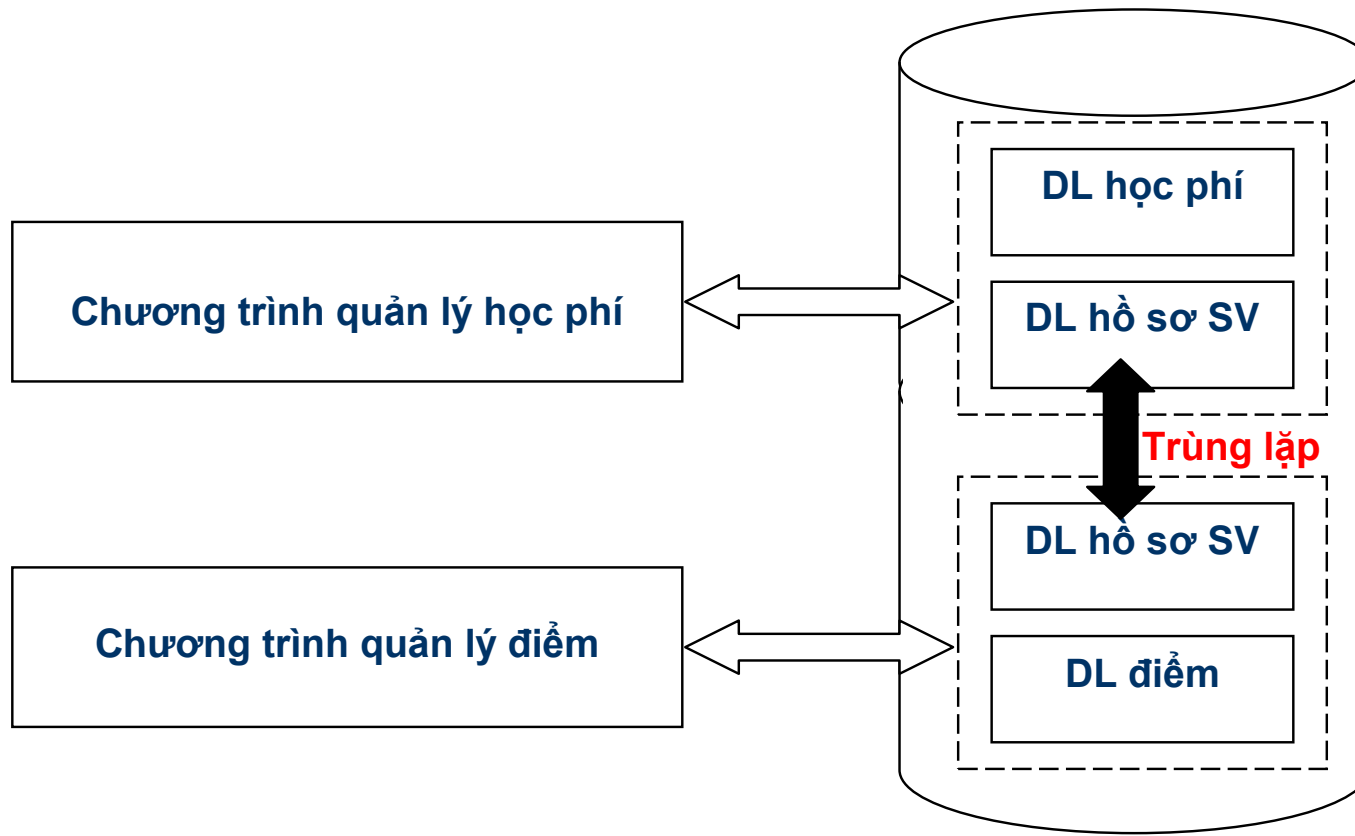
- Một phương pháp PT&TK là sự hợp thành của **3 yếu tố**:
 - Các khái niệm và mô hình
 - Quy trình thực hiện: Các bước đi lần lượt, các hoạt động cần làm
 - Công cụ trợ giúp: Phần mềm giúp (hỗ trợ) việc phân tích và thiết kế HT
- Có 02 phương pháp tiếp cận PT&TK HTTT phổ biến
 - Phương pháp PTTK hướng chức năng (Functional system analysis and design)
 - Phương pháp PTTK hướng đối tượng (Object-oriented system analysis and design)

Tiếp cận Phương pháp phân tích thiết kế HTTT

- Phương pháp PTTK hướng chức năng
 - Ra đời vào những năm 70, 80 của thế kỷ XX
 - **Lấy chức năng làm đơn vị phân rã khi tiến hành PTTK HT**
 - Cài đặt HT bằng các ngôn ngữ lập trình thủ tục (procedural programming language): Pascal, C, ...
 - Nhược điểm: **HT khó sửa chữa, khó nâng cấp, khó tái sử dụng**
- Phương pháp PTTK hướng đối tượng
 - Ra đời vào những năm 90 của thế kỷ XX
 - **Lấy đối tượng làm đơn nguyên cơ bản của HT**
 - Đối tượng: kết hợp cả chức năng và dữ liệu
 - Cài đặt bằng các ngôn ngữ lập trình hướng đối tượng (object-oriented programming language): C++, Java, C#,...

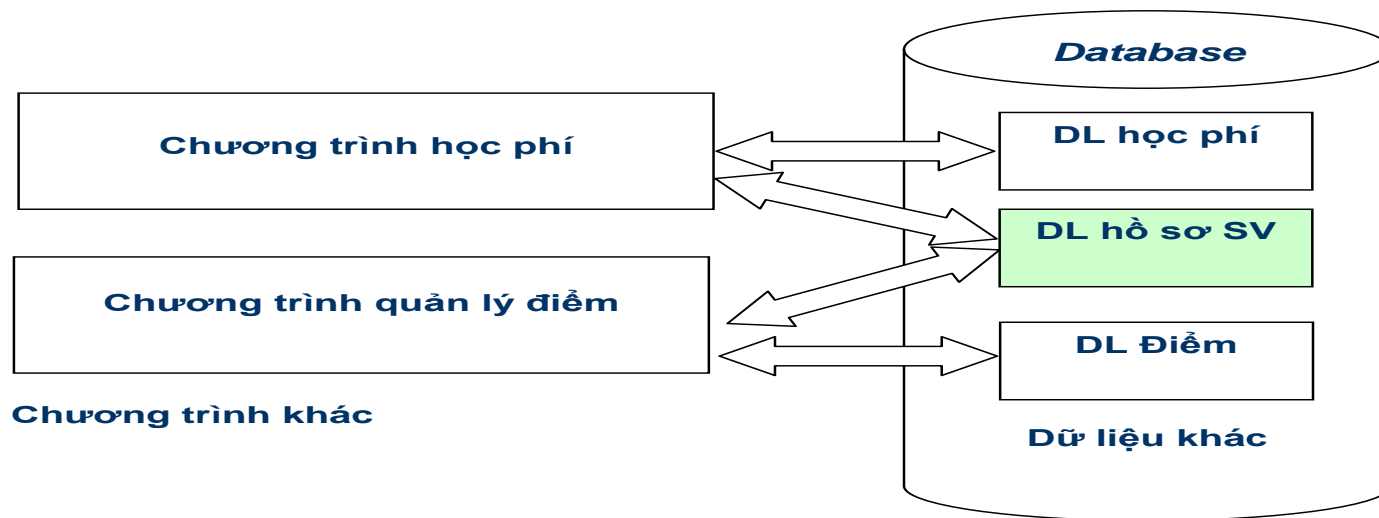
2.3. Tiếp cận hệ thống hướng chức năng

- Quan tâm các tiến trình chức năng của hệ thống



2.3. Tiếp cận hệ thống hướng chức năng

- Hạn chế của tiếp cận hướng tiến trình
 - Dữ liệu mỗi chương trình ứng dụng độc lập nhau → không thể sử dụng chung dữ liệu vì cấu trúc dữ liệu trong mỗi chương trình là khác nhau
 - Khi tiến trình thay đổi → phải tổ chức lại các tệp dữ liệu tương ứng
- Tiếp cận định hướng dữ liệu (data-oriented approach)
 - Tách dữ liệu ra khỏi các xử lý
 - Tổ chức cơ sở dữ liệu chung cho các hệ ứng dụng



2.3. Tiếp cận hệ thống hướng chức năng

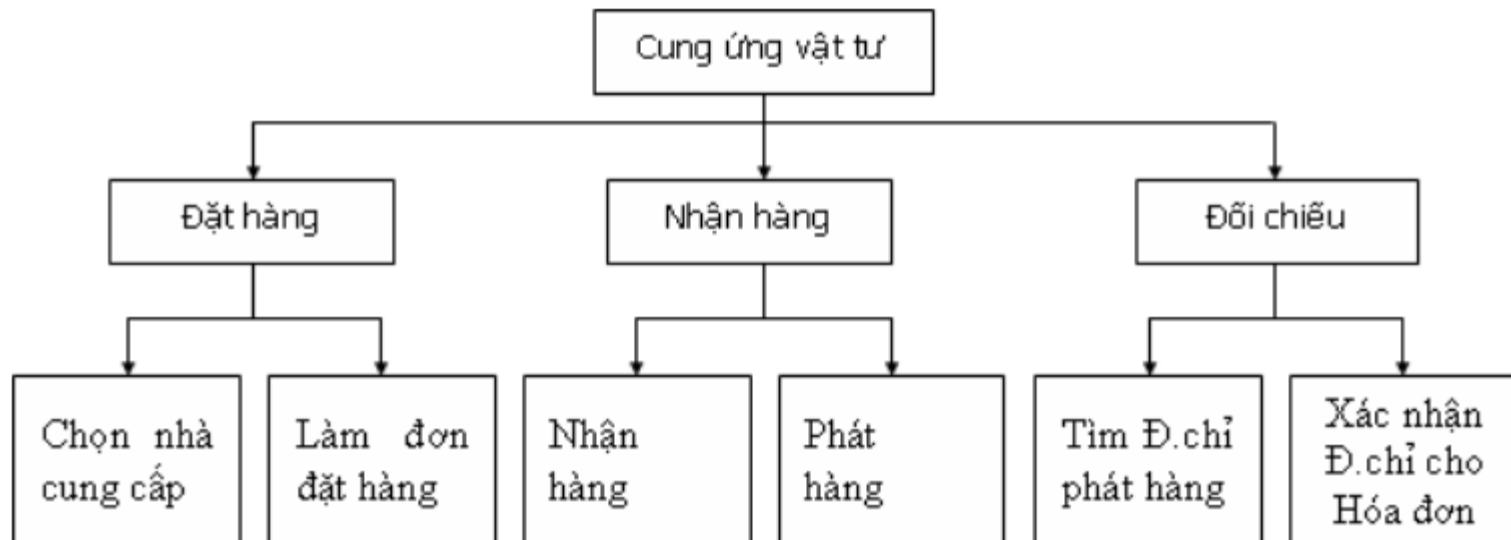
- Tiếp cận theo hướng cấu trúc (Structure - oriented)
 - Còn gọi là *tiếp cận theo hướng dữ liệu/chức năng*
 - hướng vào việc cải tiến cấu trúc các chương trình theo hướng **mô đun hoá** để dễ theo dõi, quản lý và bảo trì
 - sử dụng một số công cụ để xác định **luồng thông tin** và các **quá trình xử lý**. Việc xác định và chi tiết hoá dần các **luồng dữ liệu** và các **tiến trình** là ý tưởng cơ bản của phương pháp luận (Top - Down).
- Ưu điểm:
 - Làm giảm sự phức tạp (nhờ chia nhỏ, mô đun hoá)
 - Tập trung vào ý tưởng (vào logic, kiến trúc trước khi thiết kế)
 - Chuẩn mực hoá (theo phương pháp, công cụ đã cho)
 - ~~Hướng về tương lai (kiến trúc tốt, mô đun hoá dễ bảo trì)~~

2.3. Tiếp cận hệ thống hướng chức năng

- Tiếp cận theo hướng cấu trúc (Structure - oriented)
 - Biểu đồ phân cấp chức năng
 - Biểu đồ luồng dữ liệu
 - Biểu đồ luồng dữ liệu mức ngữ/khung cảnh
 - Biểu đồ luồng dữ liệu mức đỉnh
 - Biểu đồ luồng dữ liệu mức dưới đỉnh
 - Cơ sở dữ liệu





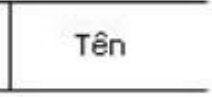

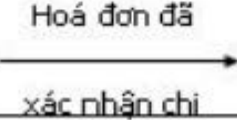
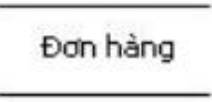
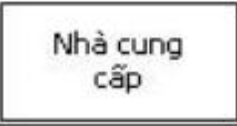
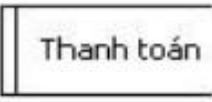
2.4. Tiếp cận hệ thống hướng chức năng

- Biểu đồ phân cấp chức năng
 - Mô tả chức năng hệ thống thông tin từ mức tổng quát đến mức chi tiết hơn
 - Chức năng là động từ

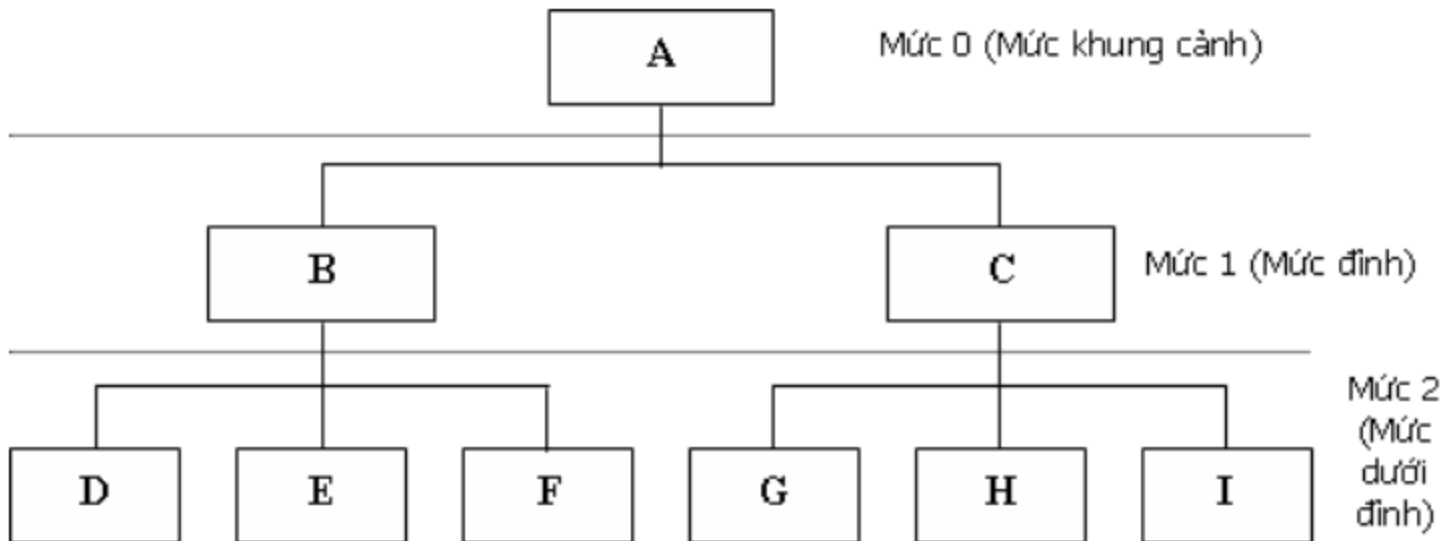


2.4. Tiếp cận hệ thống hướng chức năng

■ Biểu đồ luồng dữ liệu

	Chức năng	Luồng dữ liệu	Kho dữ liệu	Tác nhân ngoài	Tác nhân trong
Định nghĩa	Nhiệm vụ xử lý thông tin	Thông tin vào / ra một chức năng xử lý	Nơi lưu trữ thông tin trong một thời gian	Người hay tổ chức ngoài hệ thống có giao tiếp với hệ thống	Một chức năng hay một hệ con của hệ thống nhưng được mô tả ở trang khác
Tên đi kèm	Động từ (+ bổ ngữ)	Danh từ (+ tính từ)	Danh từ (+ tính từ)	Danh từ	Động từ
Biểu đồ					
Ví dụ					

2.4. Tiếp cận hệ thống hướng chức năng



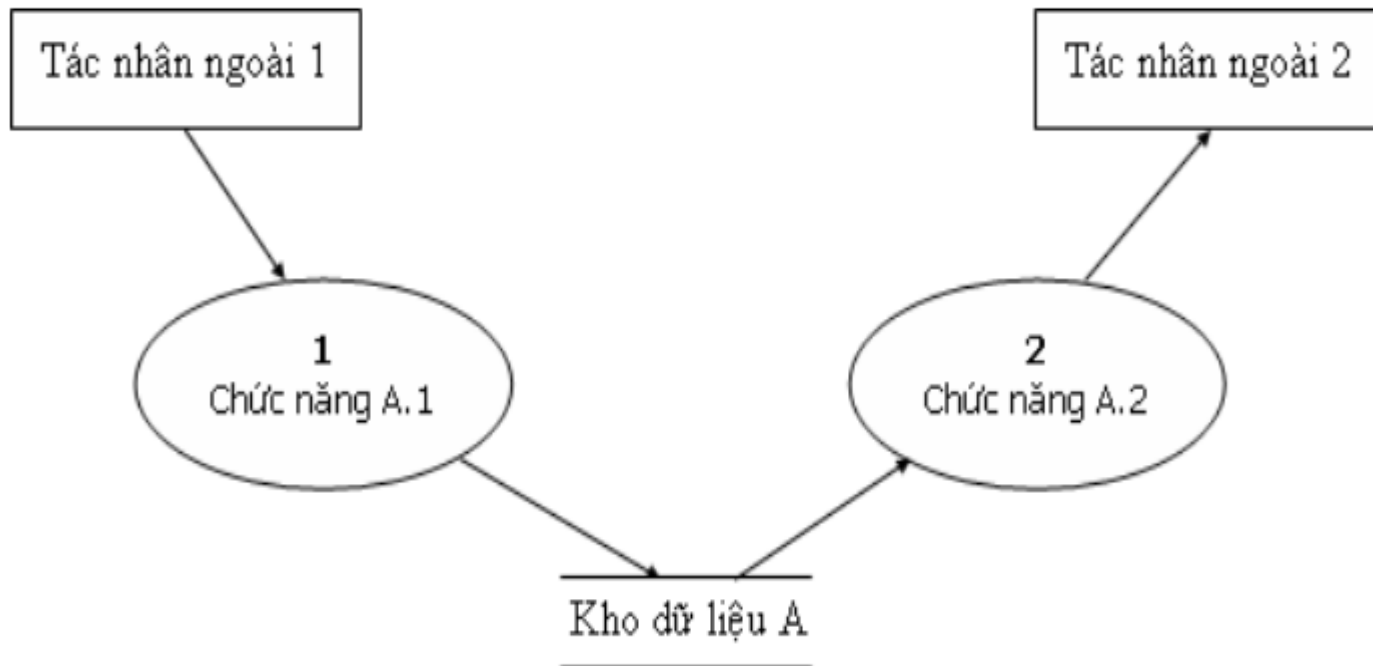
2.4. Tiếp cận hệ thống hướng chức năng

- Mức khung cảnh



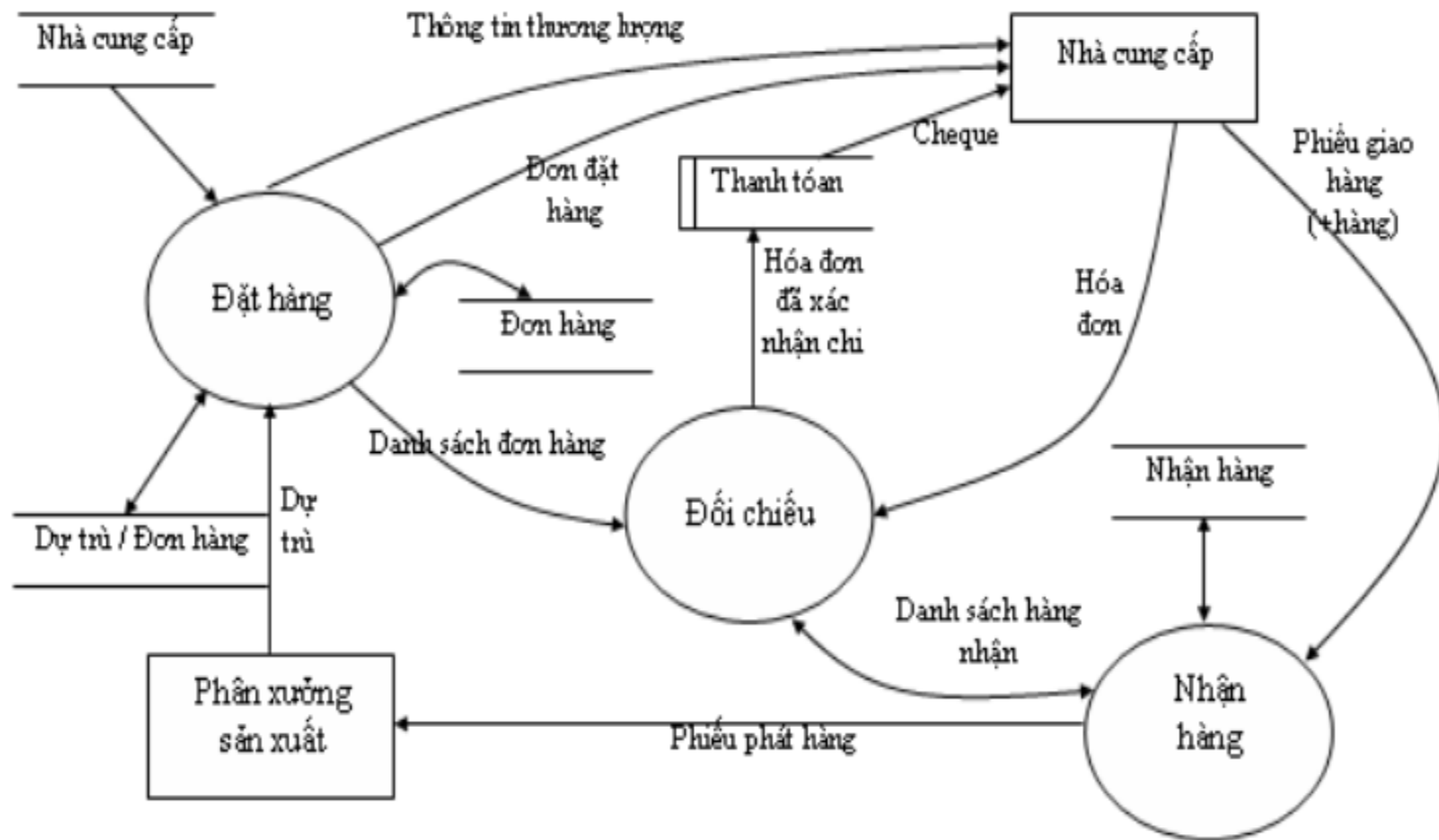
2.4. Tiếp cận hệ thống hướng chức năng

- Mức đỉnh



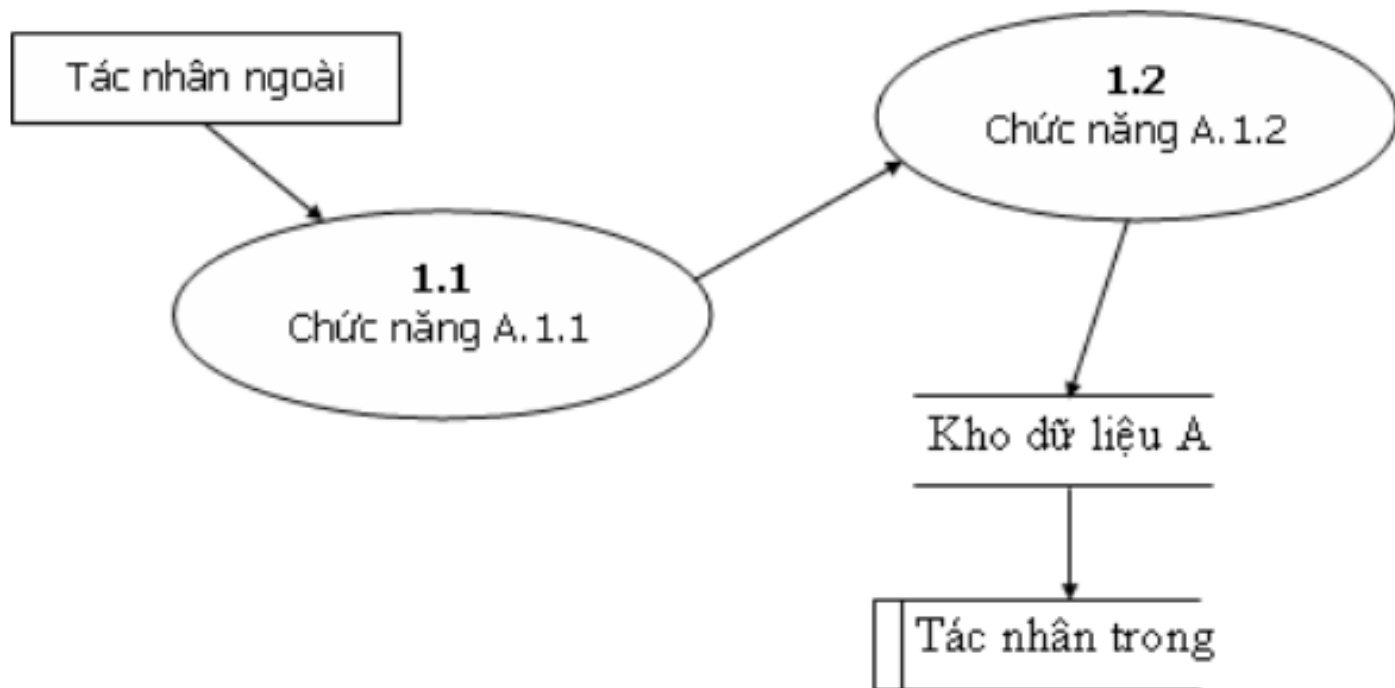
2.4. Tiếp cận hệ thống hướng chức năng

■ Mức đỉnh



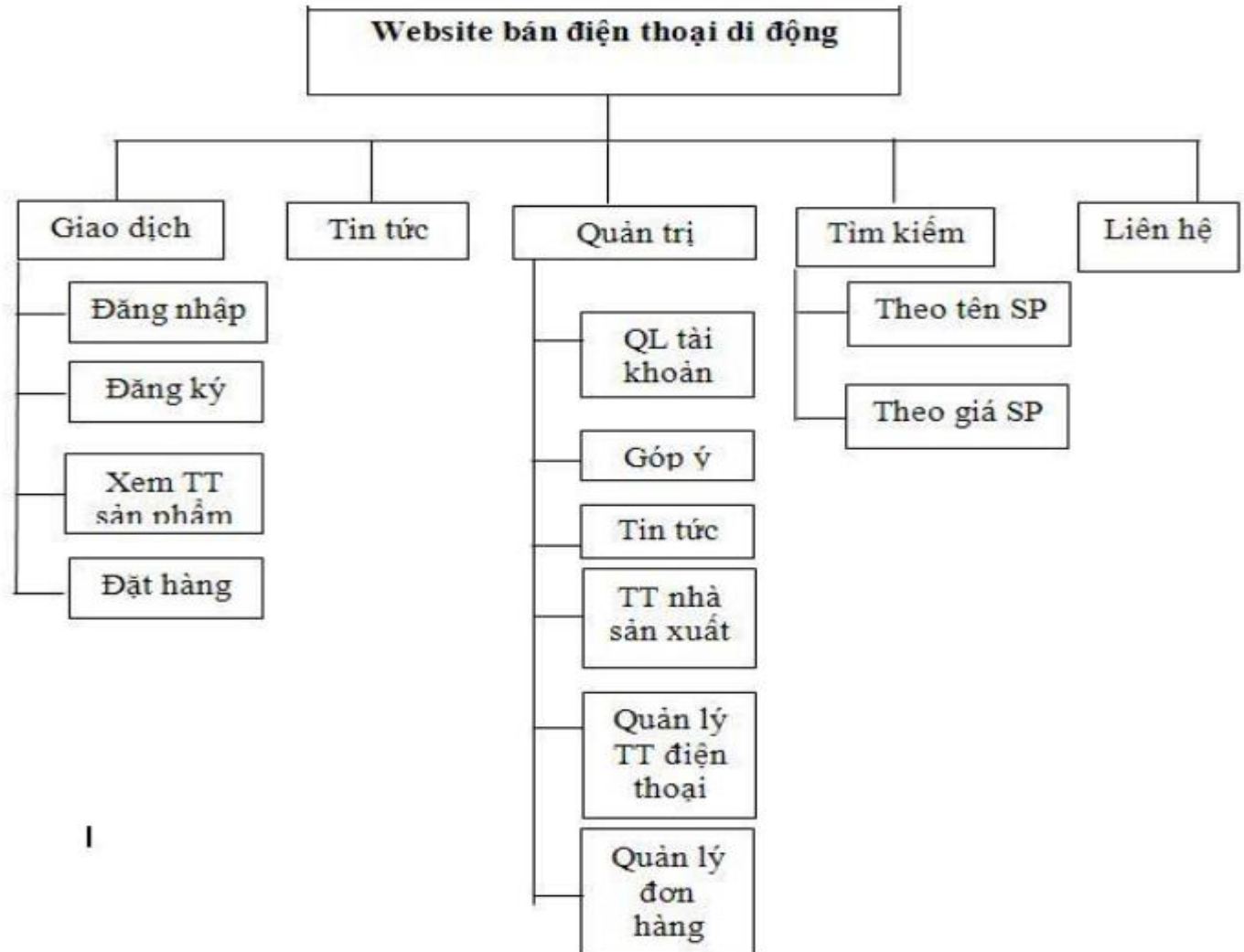
2.4. Tiếp cận hệ thống hướng chức năng

- Mức dưới đỉnh



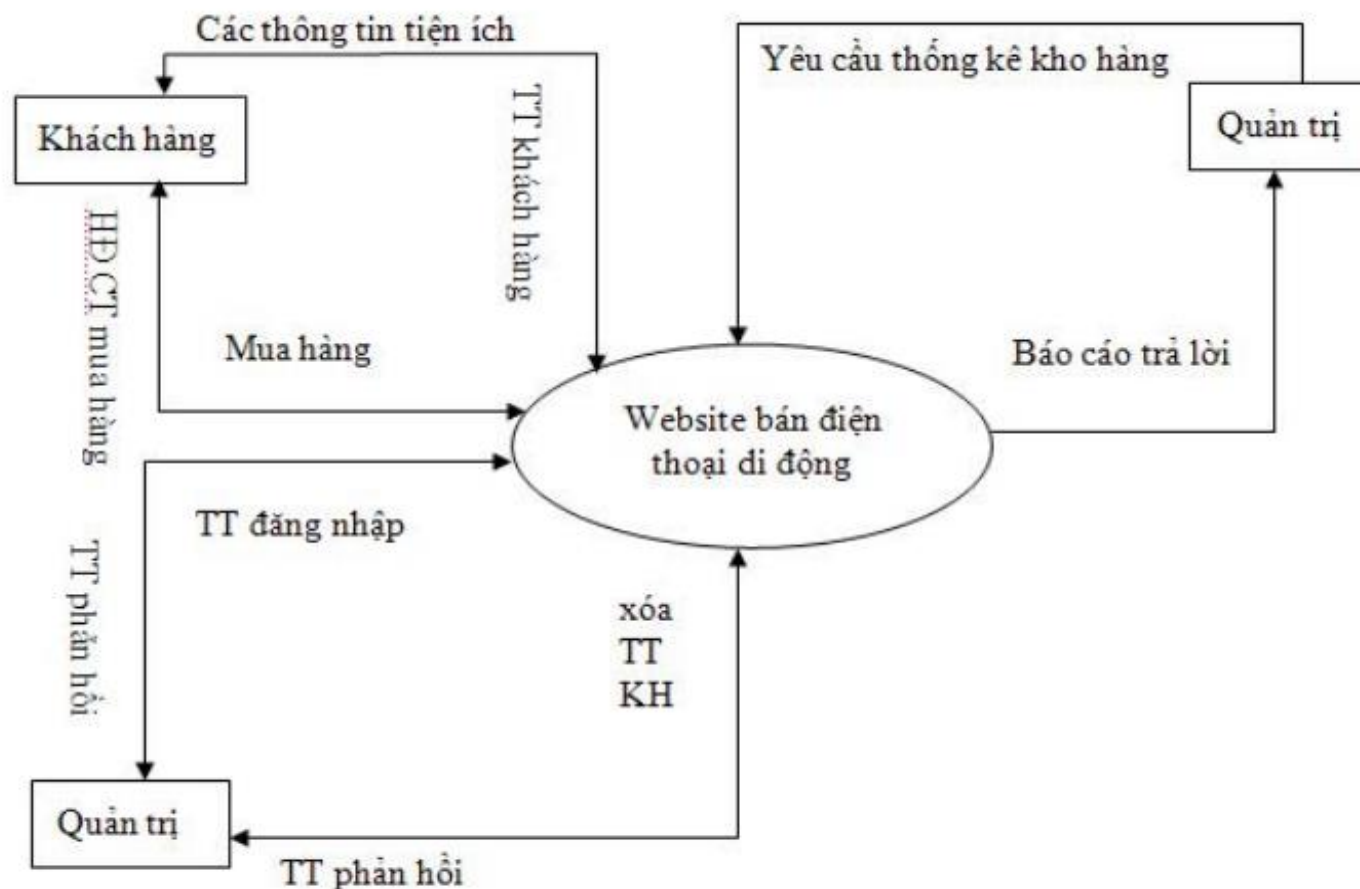
2.4. Tiếp cận hệ thống hướng chức năng

■ Ví dụ:



Biểu đồ phân cấp chức năng của hệ thống

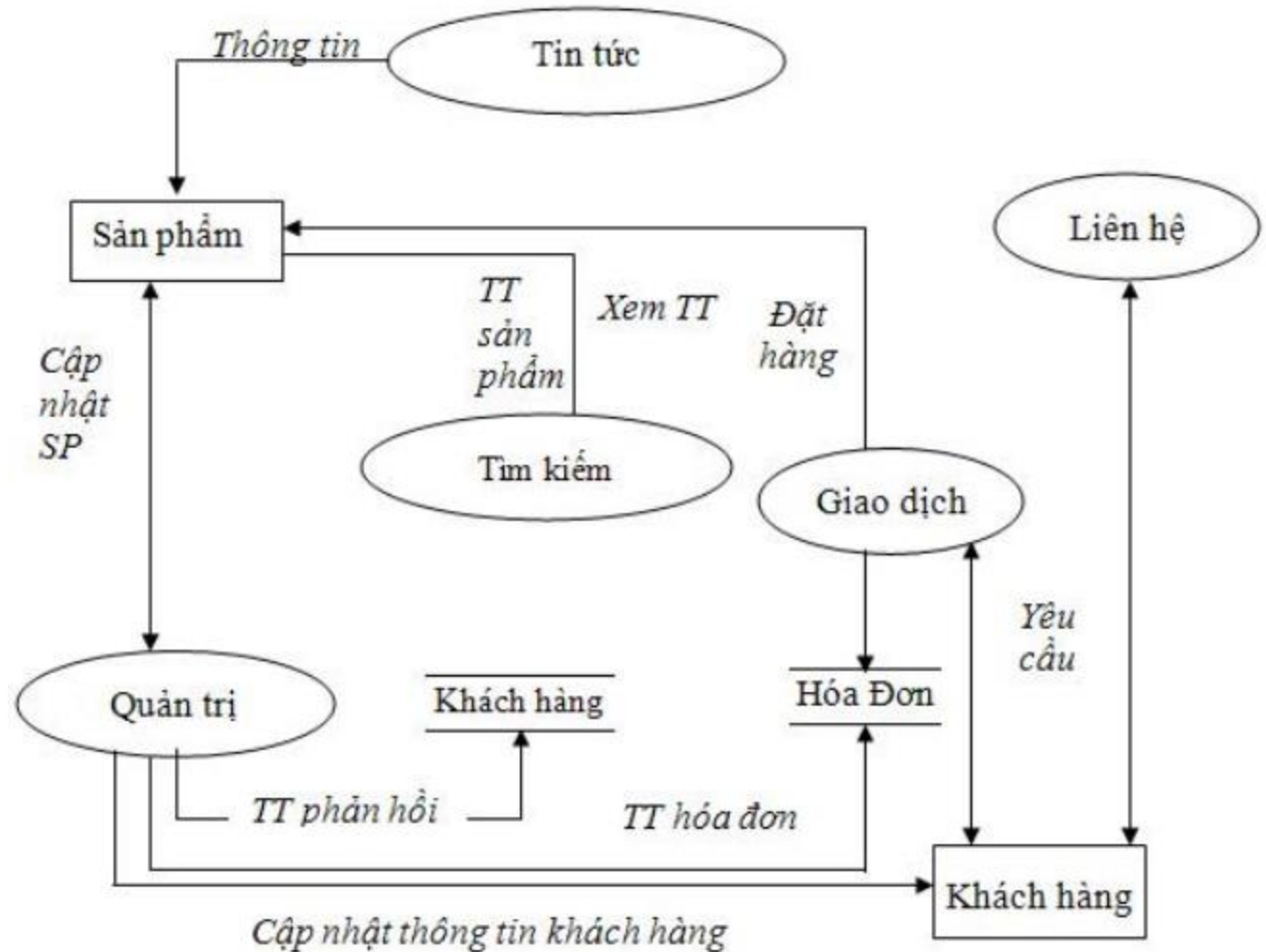
2.4. Tiếp cận hệ thống hướng chức năng



Biểu đồ luồng dữ liệu mức khung cảnh

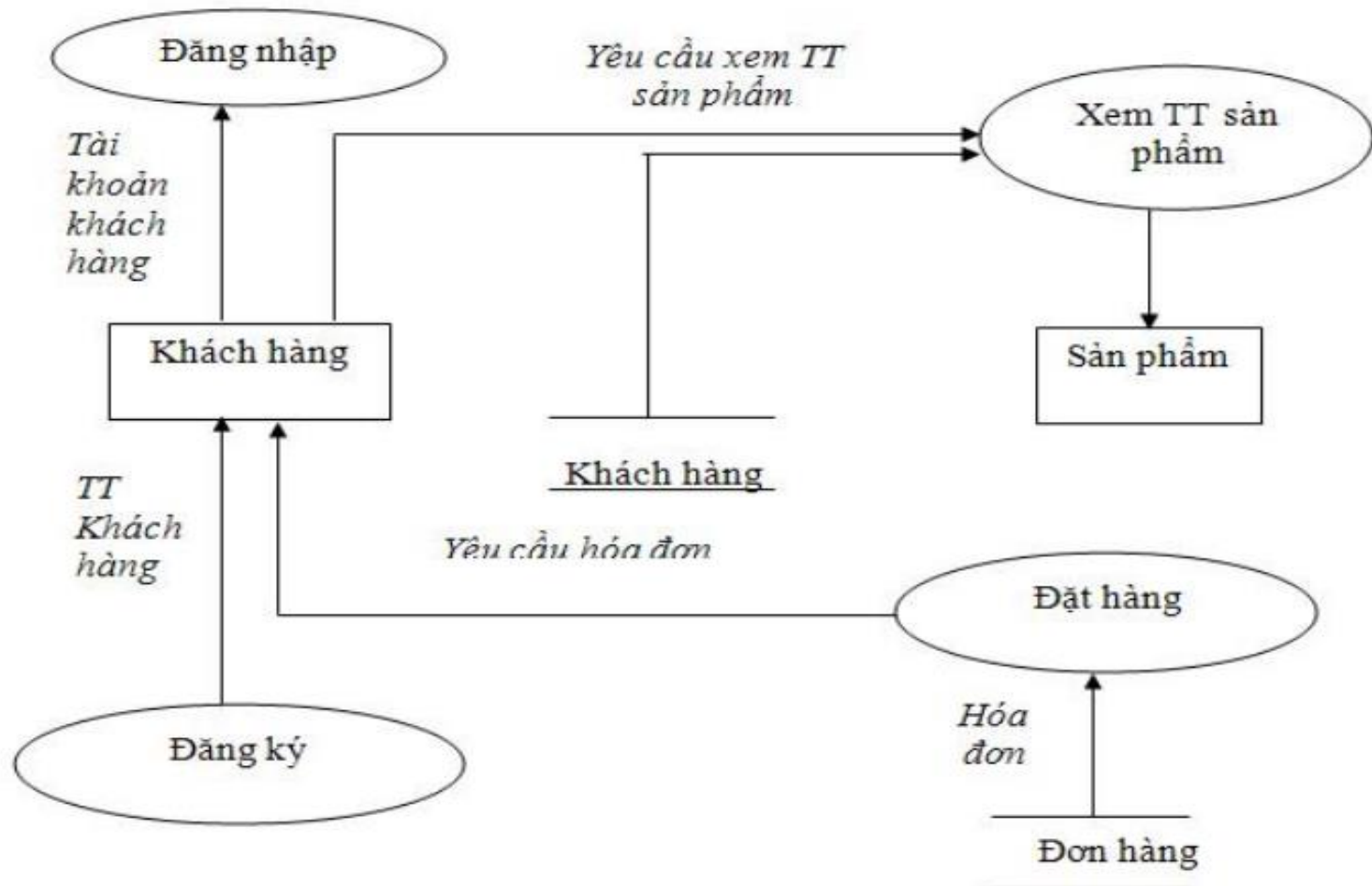
2.4. Tiếp cận hệ thống hướng chức năng

- Mức đỉnh



2.4. Tiếp cận hệ thống hướng chức năng

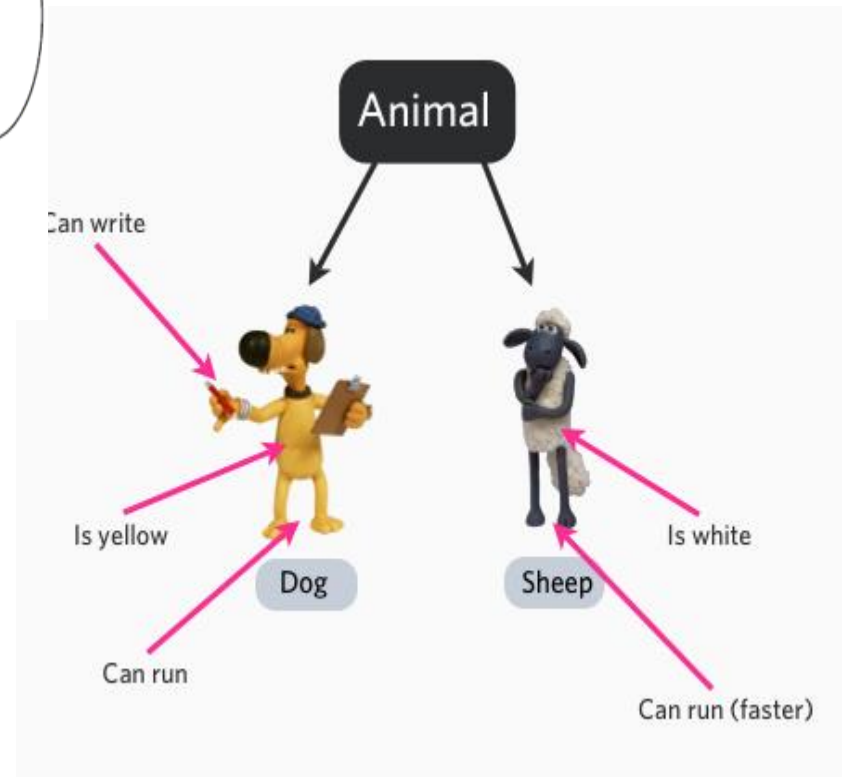
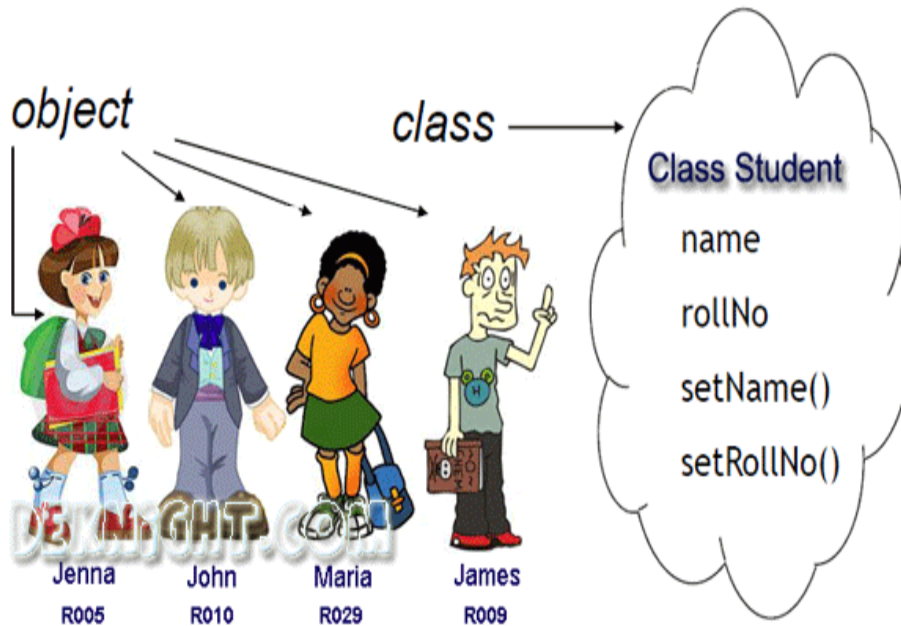
- Mức dưới đỉnh chức năng giao dịch



2.4. Tiếp cận hệ thống hướng đối tượng

- Dựa trên ý tưởng xây dựng một hệ thống gồm các **thực thể (đối tượng)** liên kết với nhau bằng các **hành vi** (quan hệ truyền thông gửi, nhận các thông báo)
 - Các đối tượng đóng gói trong nó cả dữ liệu và các xử lý
- → làm cho các phần tử của hệ thống trở nên **độc lập tương đối với nhau** và có thể **sử dụng lại**

2.4. Tiếp cận hệ thống hướng đối tượng



2.4. Tiếp cận hệ thống hướng đối tượng

- Khái niệm và mô hình hóa
 - → Mô hình hóa hướng đối tượng
- Quy trình thực hiện
 - → RUP
- Công cụ hỗ trợ
 - → UML:

Phân nhóm

- Nhóm 6-8 SV
 - Các nhóm tạo phiếu đăng ký nhóm: Ghi thông tin thành viên (**Họ tên, mã SV, ngày sinh, điện thoại, email, chữ ký...**) nhóm trưởng **chụp ảnh tờ phiếu** gửi lên hệ thống **my.vinhuni**