



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Systems development life cycle

Viet-Trung Tran

trungtv.github.io

School of Information and Communication Technology

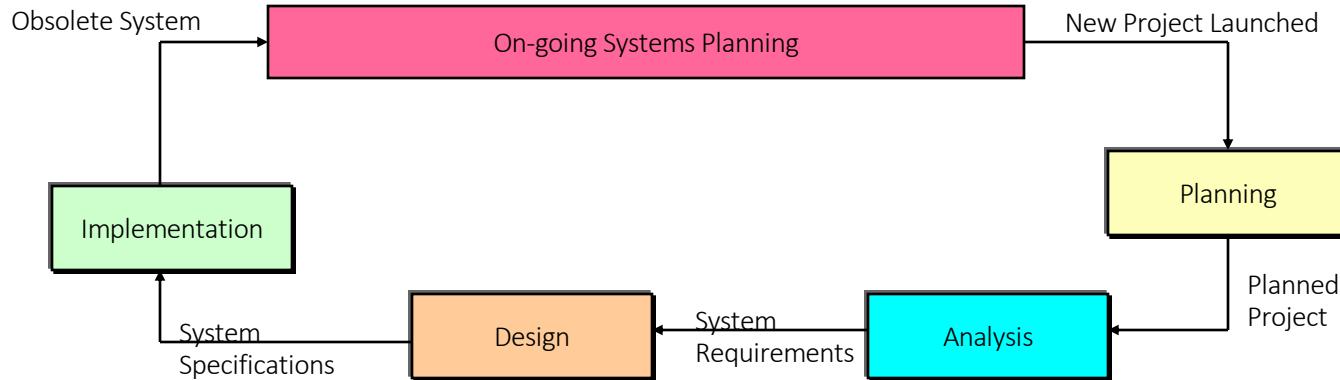
Outline

Systems development life cycle (SDLC)

- SDLC, also referred to the application development life-cycle in software engineering, is a process for planning, creating, testing, deploying, and maintaining an information system
- Should include all activities needed for
 - planning
 - systems analysis
 - systems design
 - programming
 - testing
 - user training
 - deploying
 - maintaining...

SDLC phases

- Planning
- Analysis
- Design
- Implementation



Planning Phase

- Project Initiation
 - Prepare system request
 - Perform preliminary feasibility analysis
- Set Up the Project
 - Project Plan, including work plan & staffing plan

Analysis Phase

- Determine Analysis Strategy
 - Study existing system and its problems
- Collect and Analyze Requirements
 - Develop new system concept
 - Describe new system with analysis models
- Prepare and Present System Proposal
 - Summarize results of the Analysis Phase
 - Go/No Go decision made by sponsor and steering committee

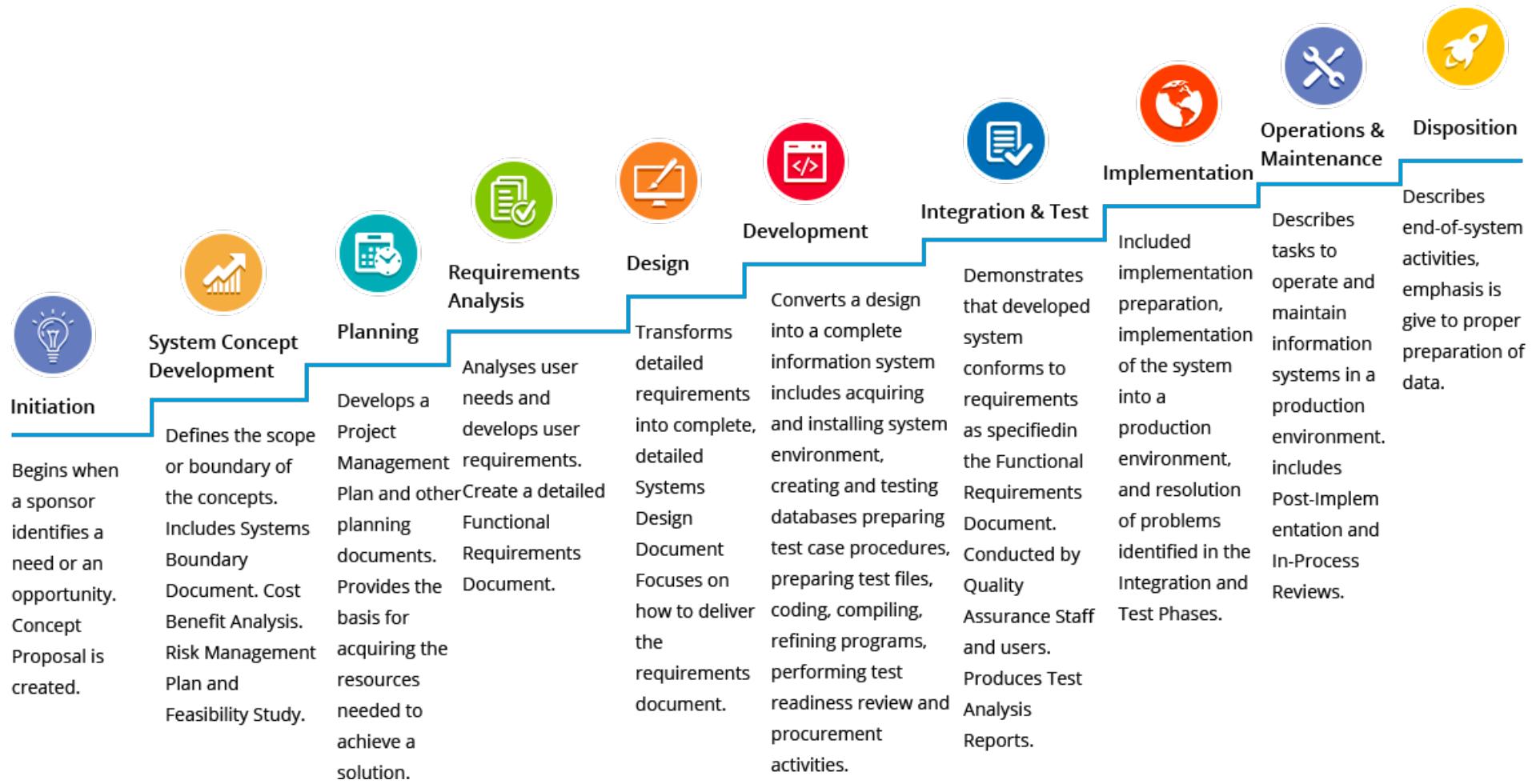
Design Phase

- Determine Design Strategy
 - Build / Buy / Outsource
- Design system components
 - Architecture, interface, database, programs
 - Assemble design elements into System Specification
- Present to steering committee
 - Go / No Go decision before entering final phase

Implementation Phase

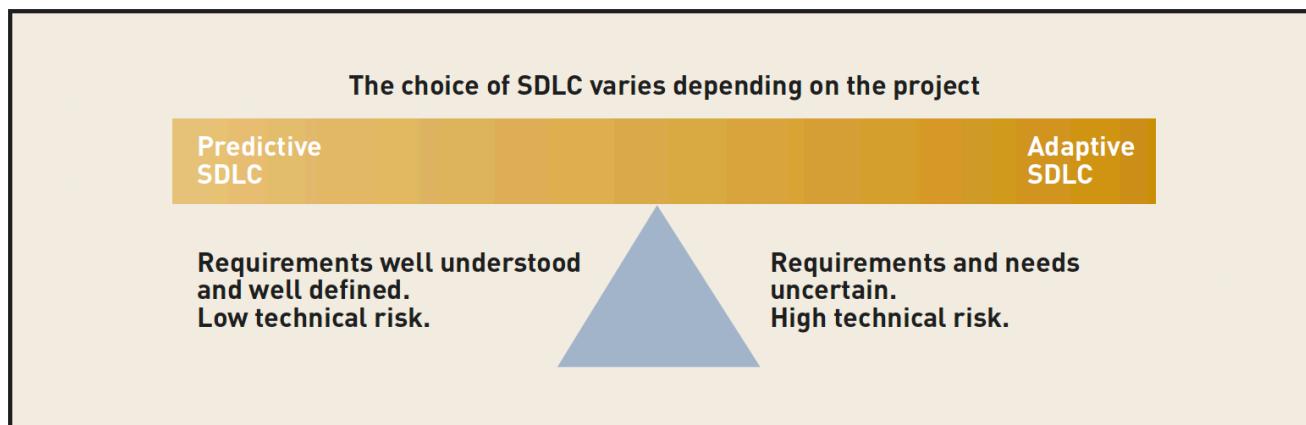
- System Construction
 - Programming and testing
- System Installation
 - Training
 - Conversion to new system
- On-going system support

A ten-phase version of the systems development life cycle



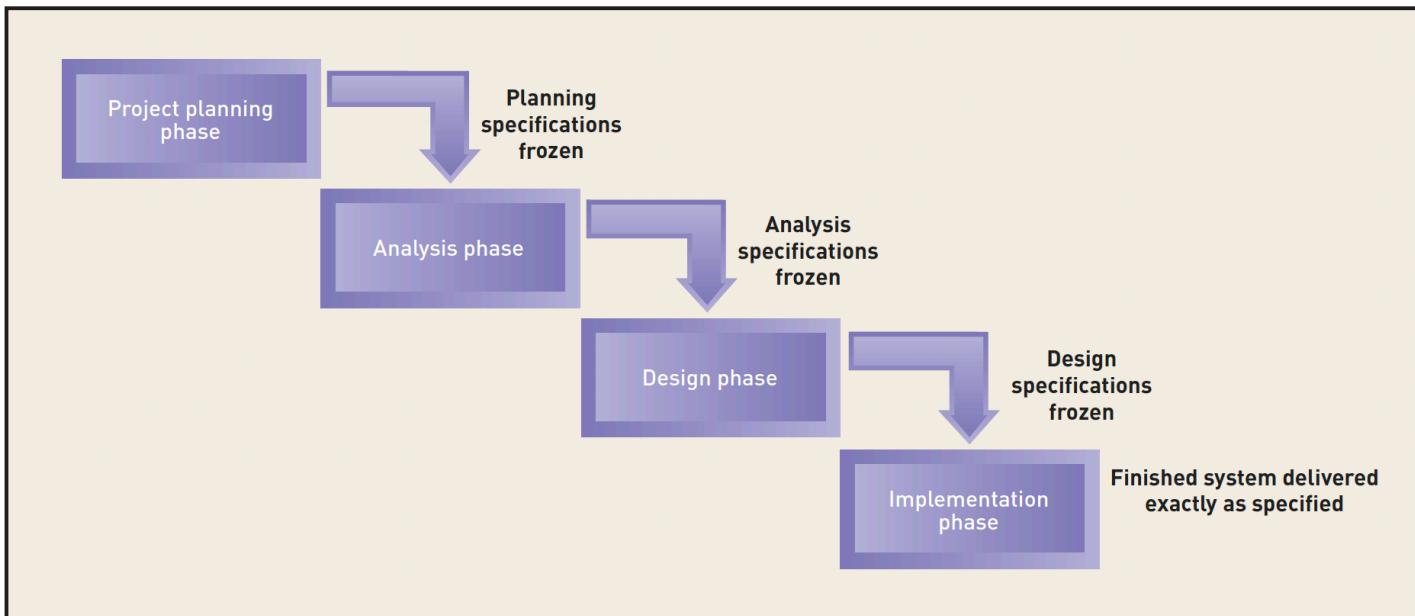
Predictive versus adaptive approaches to the SDLC

- Predictive approach assumes the development project can be planned and organized in advance and that the new information system can be developed according to the plan
- Adaptive approach is more flexible, assuming that the project cannot be planned out completely in advance but must be modified as it progresses



Waterfall model

- farthest to the left on the predictive/adaptive scale
- an SDLC approach that assumes the various phases of a project can be completed sequentially— one phase leads (falls) into the next phase



ADAPTIVE APPROACHES TO THE SDLC

- Iterative Development
 - System is “grown” in an organic way
 - Core components developed first
 - Additional components added
 - Consists of the six core development processes
 - repeated over and over until system is fully functional

Iterative development

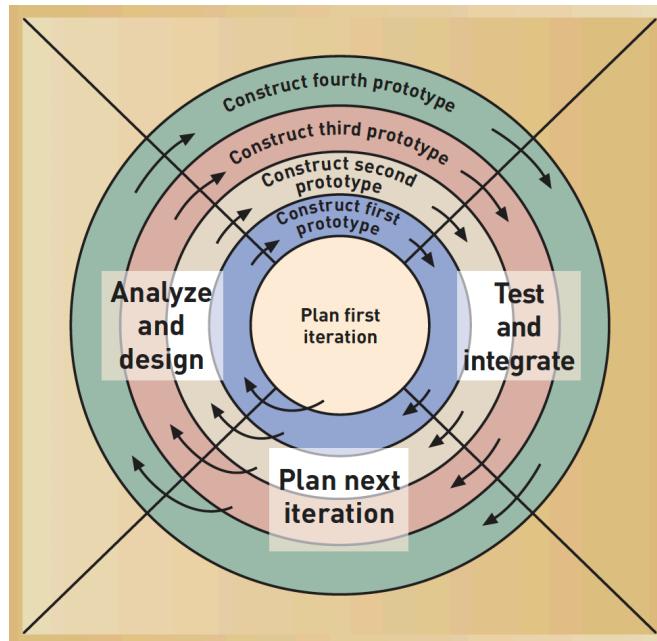
- Six Core Processes
 1. Identify problem and obtain approval
 2. Plan and monitor project
 3. Discover and understand details
 4. Design system components
 5. Build, test, integrate system components
 6. Complete system tests and deploy solution

Iterative Development

- the amount of effort expended at each iteration for each process depends on:
 - project
 - approach
 - results of previous iteration
- First iteration, more effort will go into processes 1-3
- Late iterations, more effort will go into later processes
- Each iteration should be complete in 2-4 weeks

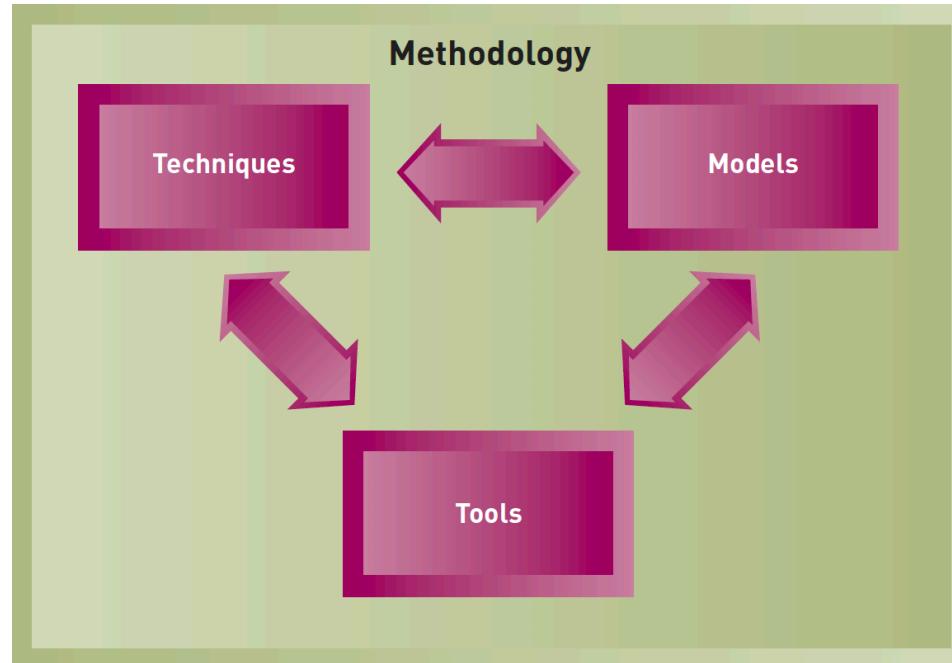
Spiral model

- Spiral model is an adaptive SDLC approach that cycles over and over again through development activities until a project
- A **spiral model** is a way to implement a **iterative model**, where each **iteration** follows a waterfall-like **model**. With each **iteration**, the product is updated, more features are added etc.



Systems Development Methodology

- A systems development methodology (or process) is a set of comprehensive guidelines for carrying out all the activities in the SDLC, including specific models, tools, and techniques
 - Some methodologies are homegrown, based on their experience
 - Some methodologies are purchased from consulting firms or other vendors



Models

- A model is a representation of an important aspect of the real world. Sometimes the term abstraction is used because we abstract (separate out) an aspect of particular importance
- Some models of system components
 - Flowchart
 - Data flow diagram (DFD)
 - Entity-relationship diagram (ERD)
 - Structure chart
 - Use case diagram
 - Class diagram
 - Sequence diagram
- Some models used to manage the development process
 - Gantt chart
 - Organizational hierarchy chart
 - Financial analysis models – NPV, ROI

Tools

- A tool in the context of system development is software support that helps create models or other components required in the project
 - Project management tools
 - Project tracking
 - Teamwork communication
 - Codebase repository
 - Visual modeling tools
 - Drawing/graphics application
 - Validate models
 - Development tools
 - Integrated development environment (IDE)
 - Word processor/text editor
 - Reverse-engineering tool
 - Code generator tool
 - Database management systems

Techniques

- A technique in system development is a collection of guidelines that help an analyst complete a system development activity or task, includes
 - step-by-step instructions for creating a model
 - general advice for collecting information from system users
- Some techniques
 - Strategic planning techniques
 - Project management techniques
 - User interviewing techniques
 - Data-modeling techniques
 - Relational database design techniques
 - Structured analysis technique
 - Structured design technique
 - Structured programming technique
 - Software-testing techniques
 - Object-oriented analysis and design techniques

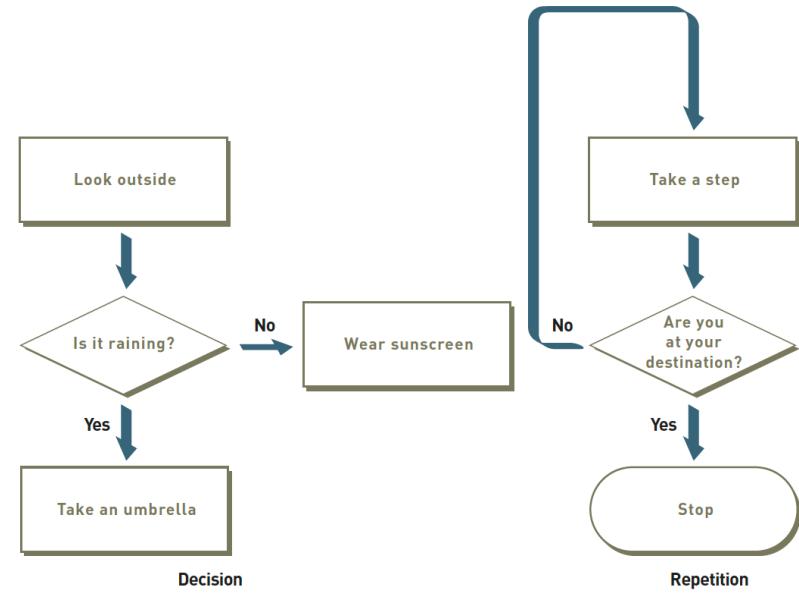
Approaches to system development

Structured system development approach

- Structured analysis, structured design, and structured programming are the three techniques that make up the structured approach
- This is the traditional approach

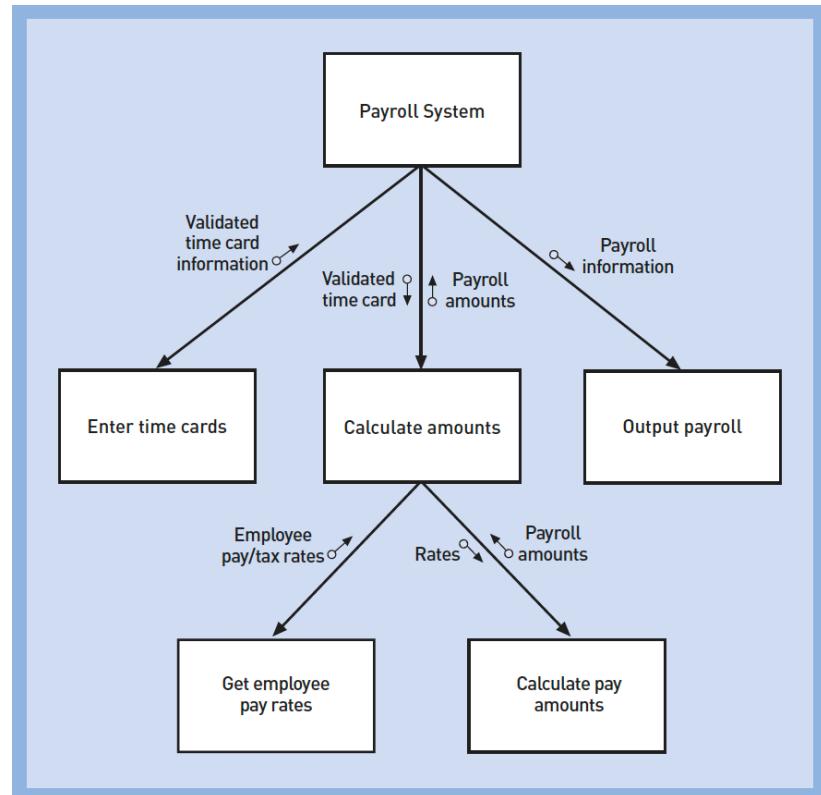
Structured programming

- A structured program is one that has one beginning and one ending, and each step in the program execution consists of one of three programming constructs
 - A sequence of program statements
 - A decision where one set of statements or another set of statements executes
 - A repetition of a set of statements



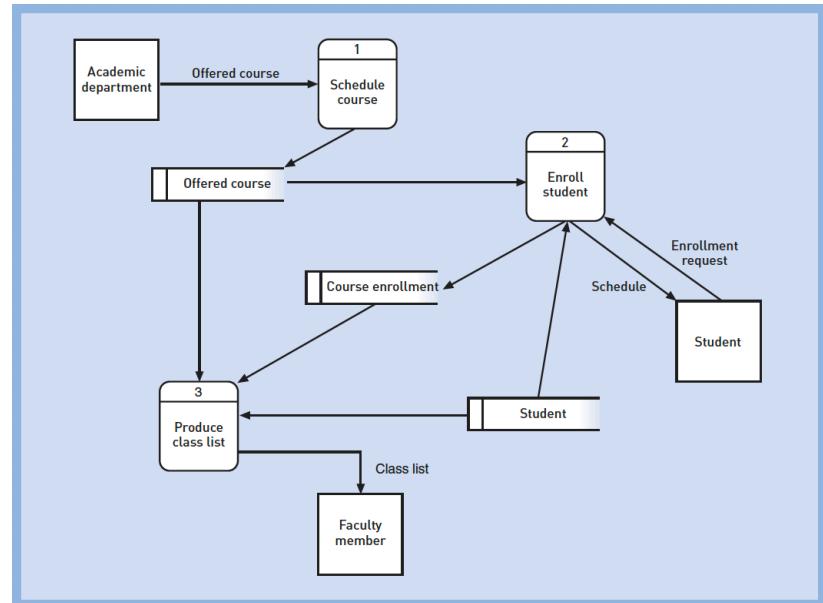
Structured Design

- A technique providing guidelines for deciding what the set of programs should be, what each program should accomplish, and how the programs should be organized into a hierarchy
- Structure chart is a graphical model showing the hierarchy of program modules produced by the structured design technique



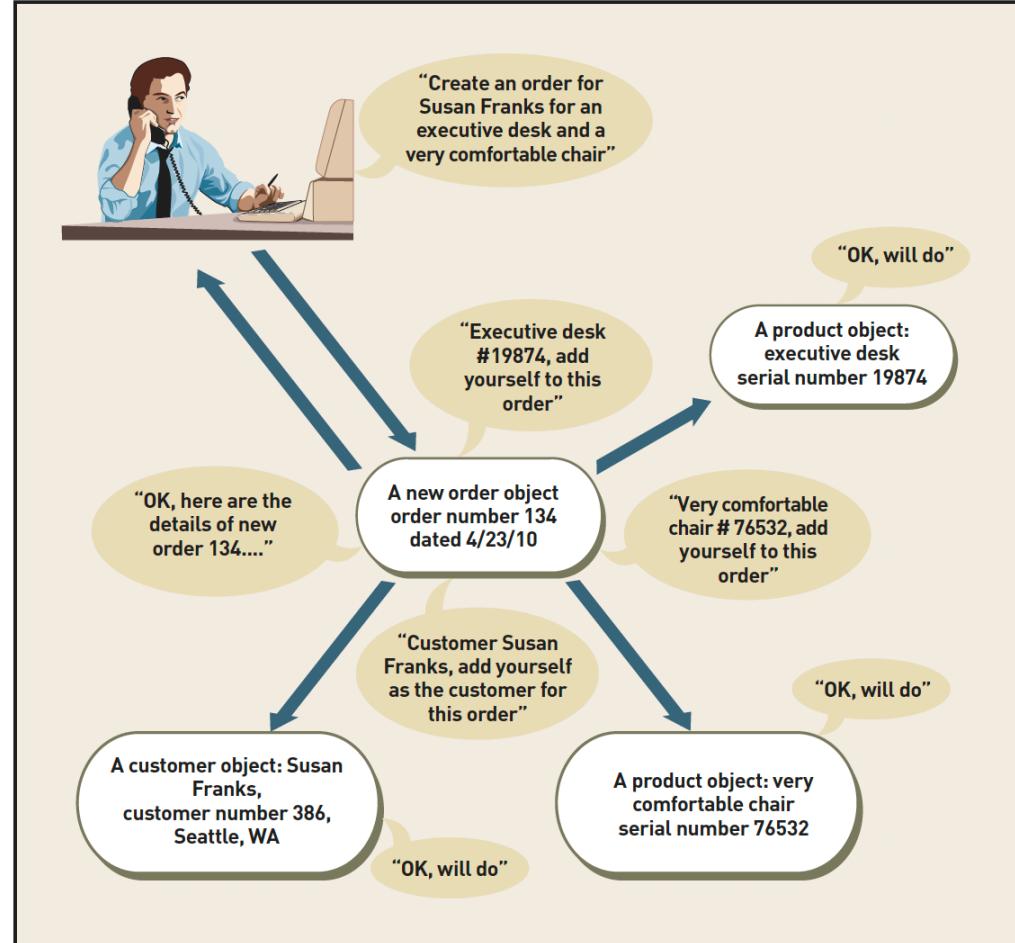
Structured analysis

- A technique used to define what processing the system needs to do, what data it needs to store and use, and what inputs and outputs are needed
- Data flow diagram (DFD) is a structured analysis model showing the inputs, processes, storage, and outputs of a system



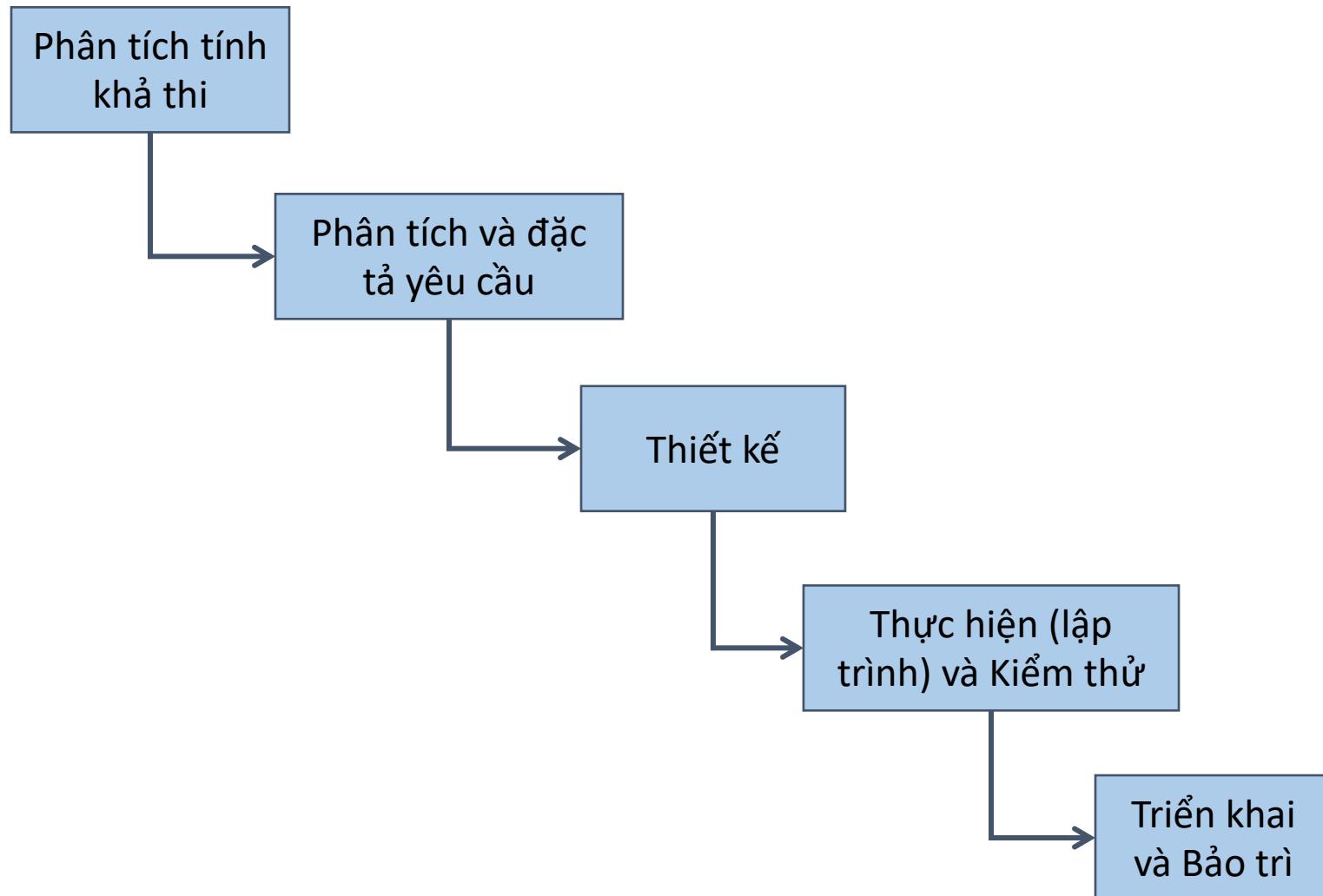
The object-oriented approach

- An approach to system development that views an information system as a collection of interacting objects that work together to accomplish tasks
- **Object** is a thing in the computer system that can respond to messages



Some SDLC in detail

Mô hình thác nước (Waterfall model)



Mô hình thác nước (Waterfall model)

- Được giới thiệu bởi Winston Royce vào năm 1970, và hiện tại vẫn là mô hình được sử dụng phổ biến nhất trong các dự án PTPM
- Việc PTPM dựa trên một tập hợp các giai đoạn (phases) có thứ tự liên tiếp
 - Trật tự (thứ tự) của các giai đoạn là xác định, và các kết quả của một giai đoạn trước sẽ được sử dụng làm đầu vào (input) cho các giai đoạn sau
- Một khi tiến trình PTPM kết thúc và hệ thống phần mềm được bàn giao (signed off) cho khách hàng, thì hệ thống phần mềm sẽ không thể được thay đổi, điều chỉnh
 - Tiến trình PTPM chỉ có thể được mở lại (để đáp ứng các điều chỉnh, thay đổi) thông qua một quy trình thực hiện thay đổi chính thức (a formal change process)
- Đặc điểm quan trọng nhất của Quy trình thác nước: **các giai đoạn (phases) không giao nhau, không lặp lại** (trong một tiến trình PTPM)
 - Giai đoạn Thiết kế (Design) không thể bắt đầu cho đến khi giai đoạn Phân tích (Analysis) được hoàn thành, và Giai đoạn Kiểm thử (Testing) không thể bắt đầu cho đến khi giai đoạn Thực hiện, lập trình (Implementation) được hoàn thành

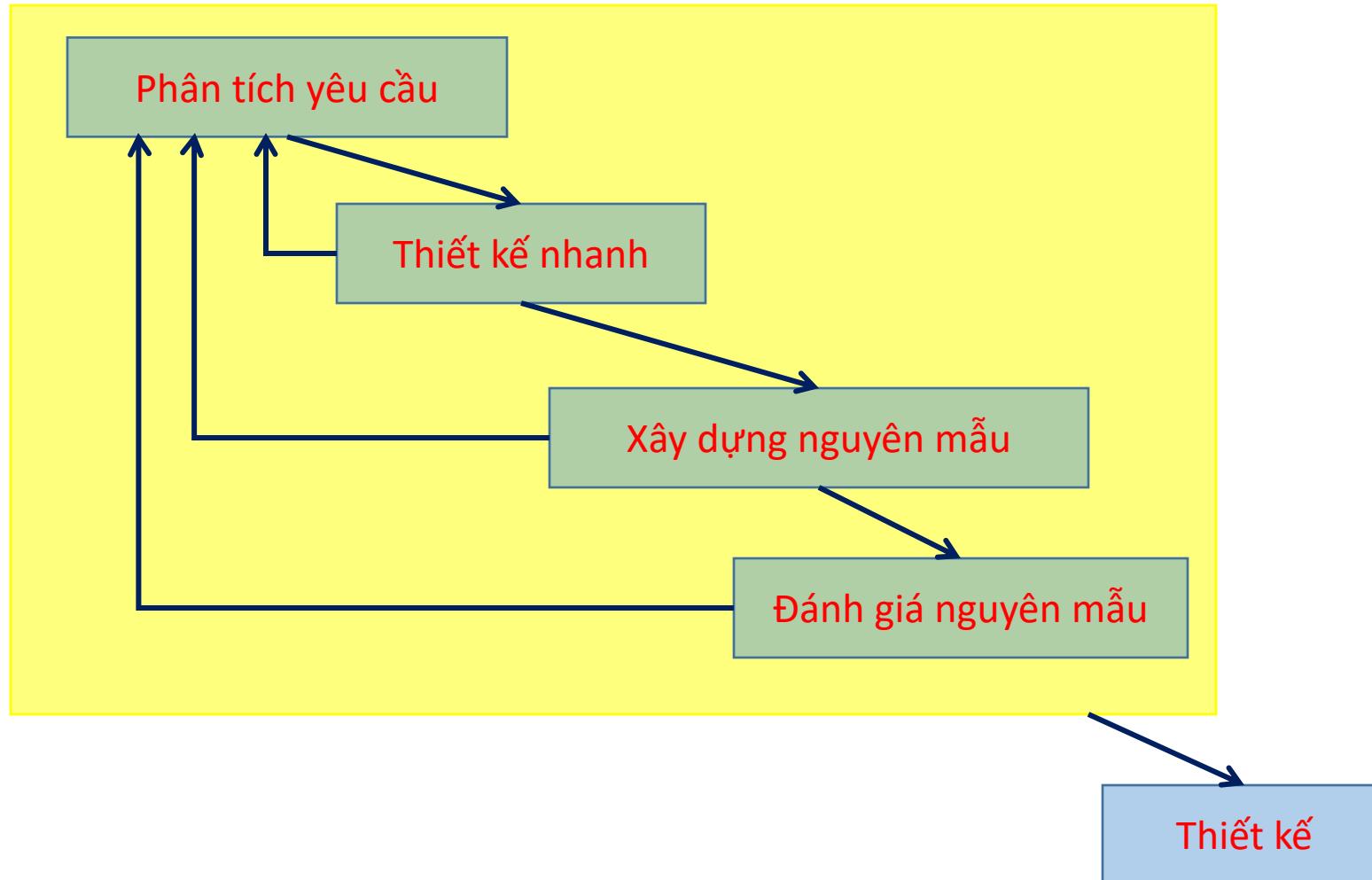
Mô hình thác nước (Waterfall model)

- Các ưu điểm
 - ❖ Là quy trình PTPM đơn giản, dễ hiểu, và dễ sử dụng
 - ❖ Các tài liệu được hoàn thành sau mỗi giai đoạn
 - ❖ Các yêu cầu phần mềm được cung cấp sớm cho các người kiểm thử (the testers)
 - ❖ Cho phép người quản lý dự án (Project Manager – PM) lập kế hoạch và kiểm soát thực hiện một cách chặt chẽ
 - ❖ Quy trình này cũng rất nổi tiếng và được biết bởi cả những người không chuyên về PTPM, giúp nó dễ dàng được dùng để trao đổi
- Các nhược điểm
 - ❖ 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** (trước giai đoạn Thiết kế)
 - ❖ 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

Mô hình thác nước (Waterfall model)

- Khi nào nên sử dụng mô hình thác nước?
 - **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)

Mô hình nguyên mẫu (Prototyping model)



Mô hình nguyên mẫu (Prototyping model)

- Thay vì cố định các yêu cầu trước khi tiến hành thiết kế hoặc thực hiện (lập trình), **một (hoặc một số các) nguyên mẫu (prototype) được xây dựng để hiểu chính xác các yêu cầu phần mềm**
- Mỗi nguyên mẫu (prototype) được xây dựng dựa trên các yêu cầu phần mềm hiện thời (thu được từ đánh giá các nguyên mẫu trước)
- Nhờ việc sử dụng thử nguyên mẫu, khách hàng có thể có được “cảm nhận thực tế” về hệ thống phần mềm, bởi vì các tương tác với nguyên mẫu cho phép khách hàng hiểu rõ hơn, chính xác hơn về các yêu cầu của hệ thống phần mềm mong muốn
- Sử dụng nguyên mẫu là hợp lý đối 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

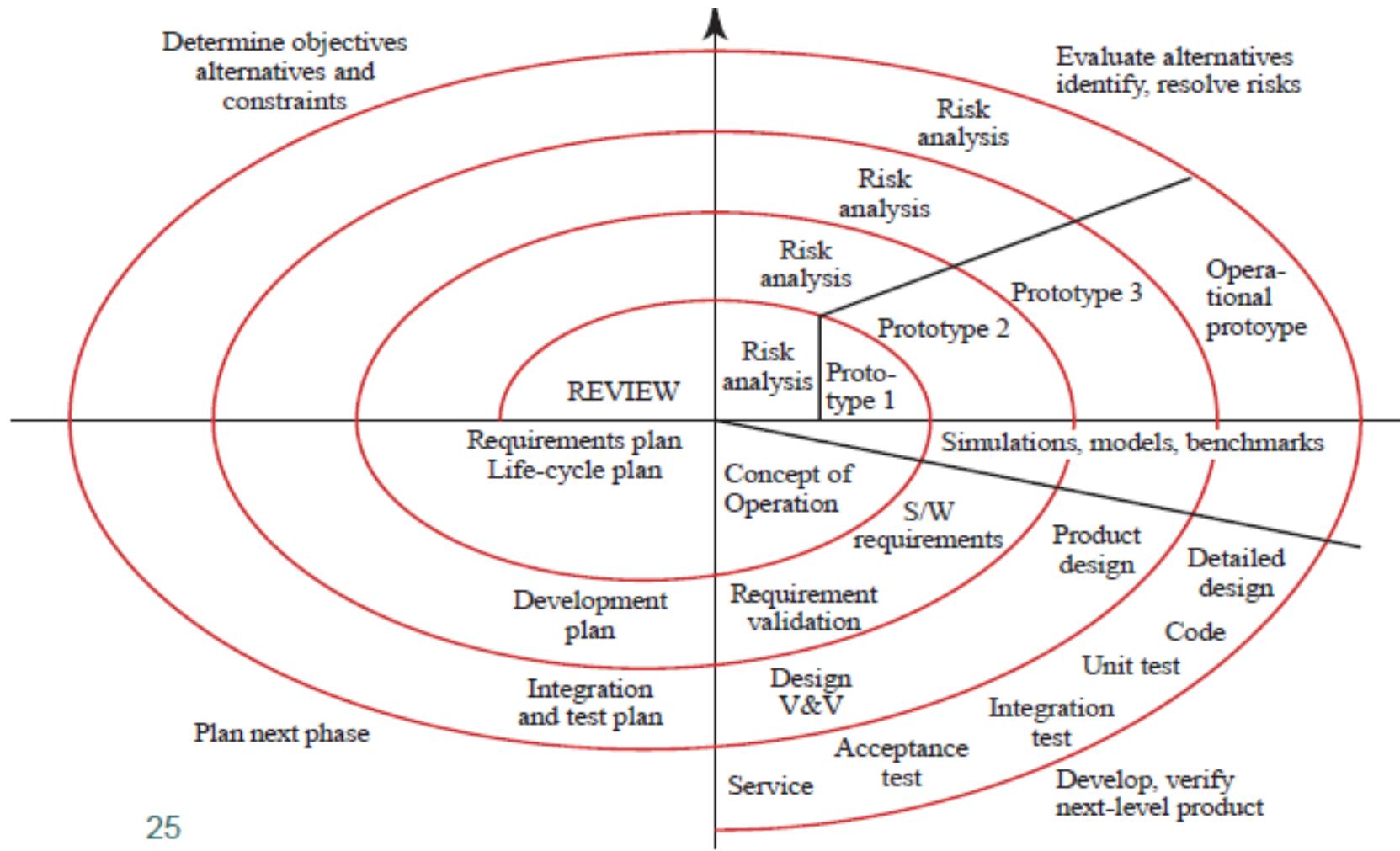
Mô hình nguyên mẫu (Prototyping model)

- Các ưu điểm
 - ❖ Người sử dụng được tham gia tích cực vào quá trình PTPM
 - ❖ Sử dụng nguyên mẫu là một mô hình hoạt động của hệ thống, những 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 có thể được phát hiện sớm
 - ❖ Các chức năng không rõ ràng hoặc khó thao tác có thể được phát hiện
- Các 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

Mô hình nguyên mẫu (Prototyping model)

- Khi nào nên dùng mô hình nguyên mẫu?
 - ❖ Khi các yêu cầu phần mềm không thể được xác định tại thời điểm bắt đầu dự án
 - ❖ **Khi 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 thu được thông qua việc dùng thử nghiệm
 - ❖ Khi khách hàng yêu cầu chứng minh tính khả thi
 - ❖ Khi cần có các demos cho các cấp quản lý ở mức cao
 - ❖ Khi các vấn đề về công nghệ cần được thử nghiệm, kiểm tra

Mô hình xoắn ốc (Spiral model)



Mô hình xoắn ốc (Spiral model)

- Được đề xuất bởi Barry Boehm
- Là một mô hình phát triển tiến hóa, dựa trên sự kết hợp lai ghép của đặc điểm phát triển lặp (iterative) của Mô hình nguyên mẫu (Prototyping model) và phát triển theo các bước tuần tự (sequential) của Mô hình thác nước (Waterfall model)
 - **Có chú trọng vào việc phân tích nguy cơ (risk analysis)**
- Trong mô hình xoắn ốc, hệ thống phần mềm được phát triển qua một chuỗi các phiên bản tăng cường (incremental releases)
 - Trong các bước phát triển lặp ban đầu, thì các phiên bản của hệ thống phần mềm có thể chỉ là các mô hình được phác thảo trên giấy hoặc là các nguyên mẫu (prototypes)
 - Trong các bước phát triển lặp về sau, thì các phiên bản ngày càng hoàn thiện của hệ thống phần mềm sẽ được tạo ra

Mô hình xoắn ốc (Spiral model)

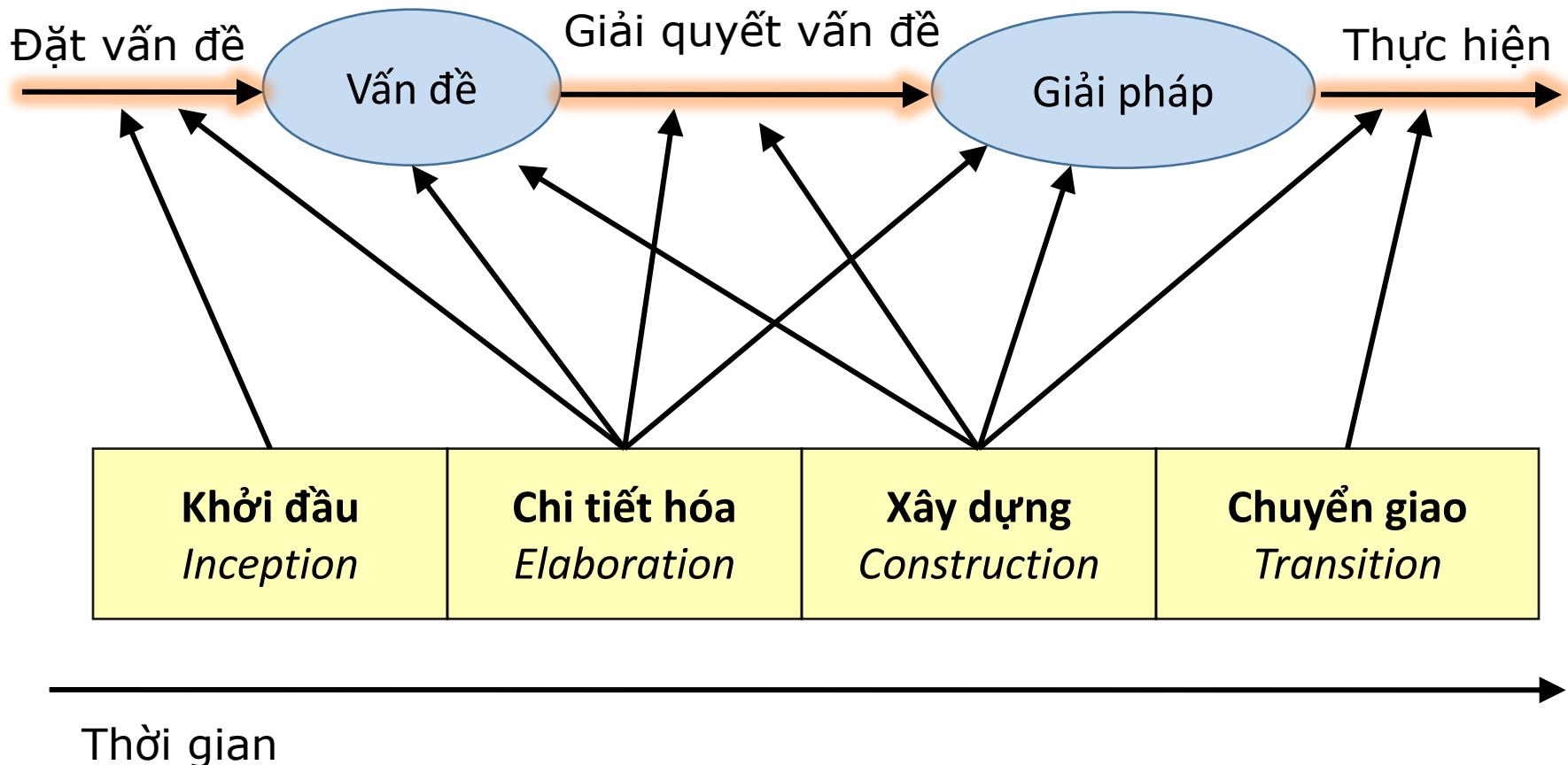
- Các ưu điểm
 - ❖ Chú trọng vào phân tích rủi ro (risk analysis), nhờ đó giúp giảm thiểu rủi ro trong dự án PTPM
 - ❖ Phù hợp đối với các dự án lớn và quan trọng đặc biệt
 - ❖ Các chức năng mới có thể được bổ sung vào sau
 - ❖ Các phiên bản đầu của hệ thống phần mềm được tạo ra sớm
- Các nhược điểm
 - ❖ Chi phí cao (thời gian, nguồn lực, tiền bạc) để áp dụng
 - ❖ Việc phân tích rủi ro (risk analysis) đòi hỏi kỹ năng và kinh nghiệm cao
 - ❖ Thành công của dự án phụ thuộc rất nhiều vào giai đoạn phân tích rủi ro (risk analysis)
 - ❖ Không phù hợp cho các dự án nhỏ

Mô hình xoắn ốc (Spiral model)

- Khi nào nên dùng mô hình xoắn ốc?
 - ❖ 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)

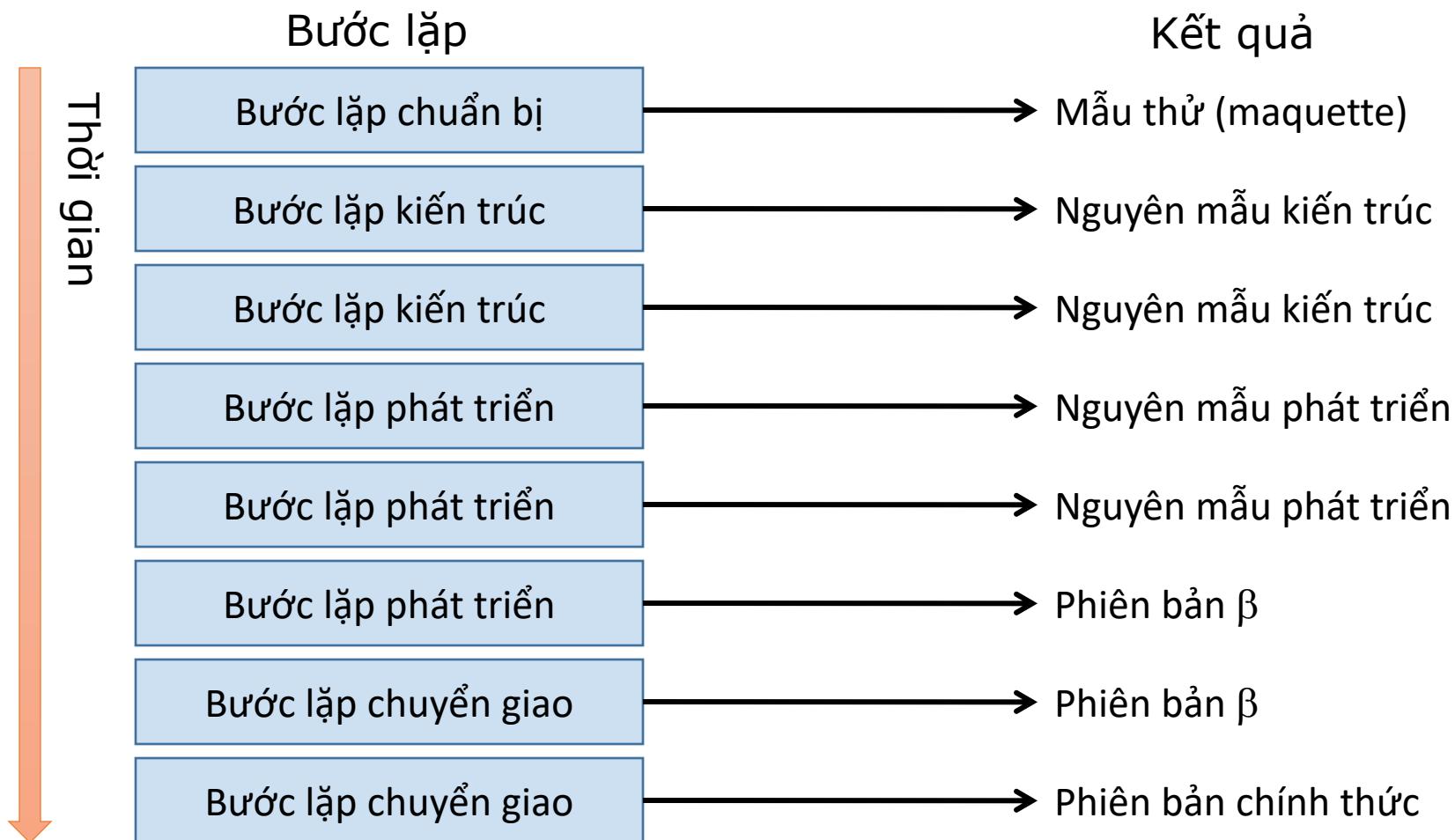
Mô hình hợp nhất (Unified model)

- Góc nhìn quản lý dự án



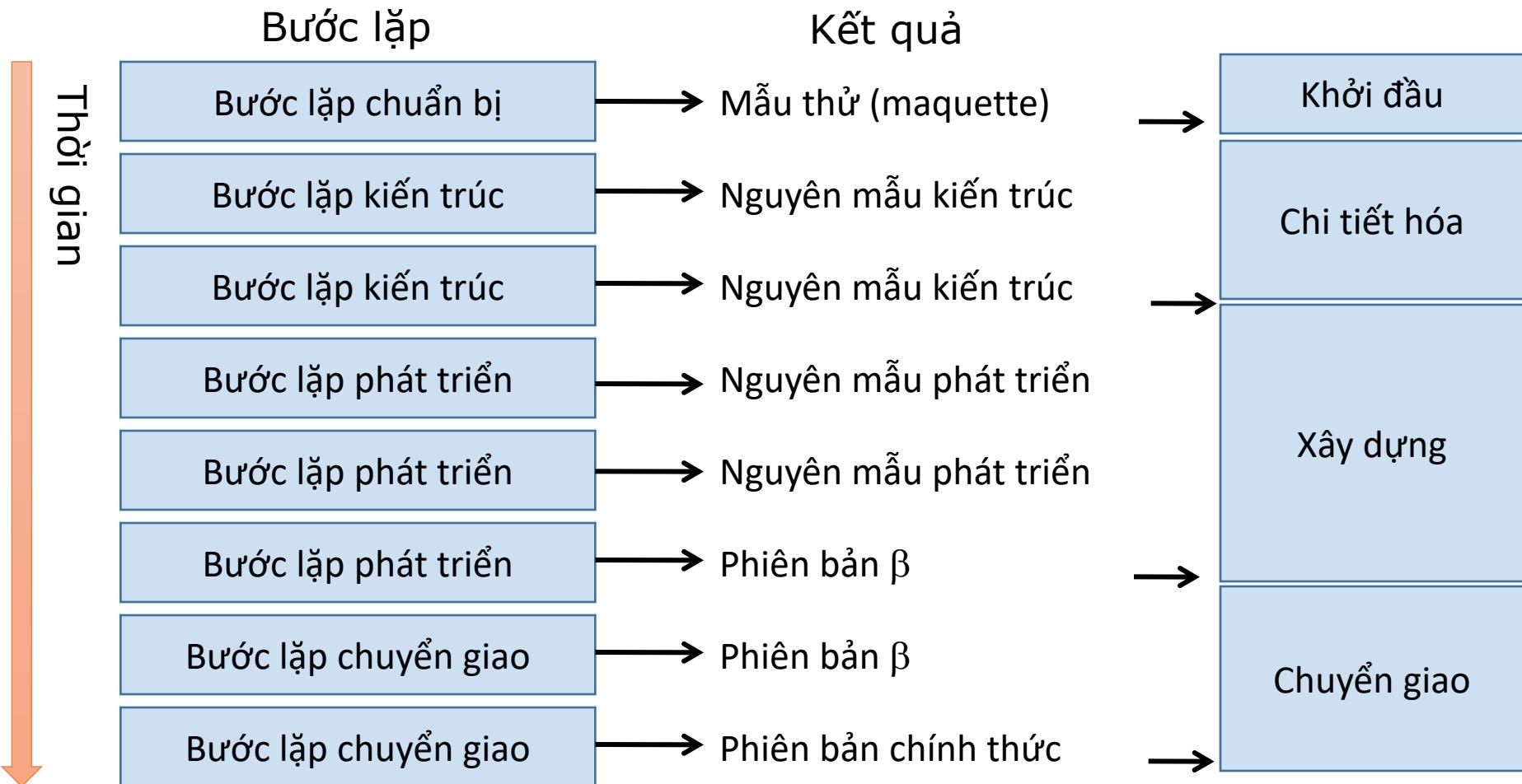
Mô hình hợp nhất (Unified model)

- Góc nhìn kỹ thuật



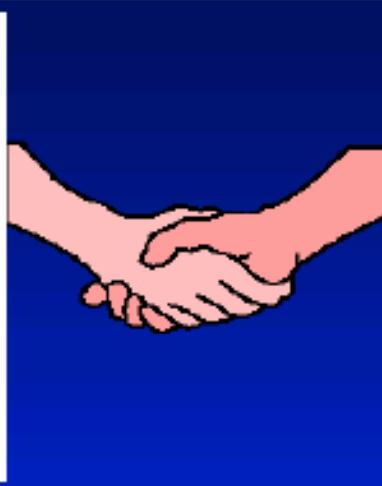
Mô hình hợp nhất (Unified model)

- Kết hợp 2 góc nhìn



Mô hình hợp nhất và UML

The Unified
Modeling
Language



The Unified
Process

Quy trình RUP

- RUP (Rational Unified Process) là một quy trình mô hình hóa với UML, không phải là một chuẩn
 - Các nguyên tắc cơ bản
 - Các giai đoạn chính (phases)
 - Các bước chính (steps)

Các nguyên tắc cơ bản (1)

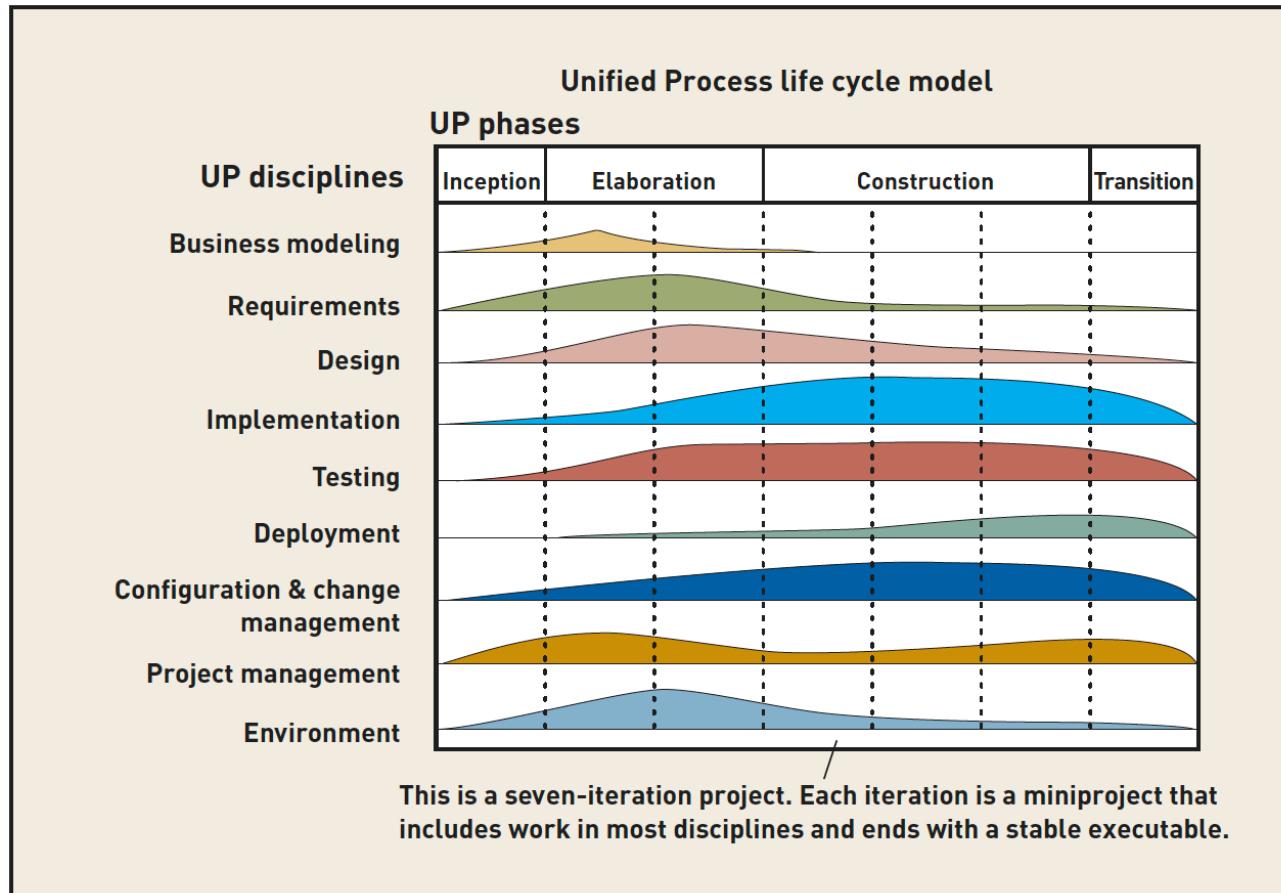
- Lắp và tăng trưởng
 - Dự án được chia thành những vòng lặp hoặc giai đoạn ngắn để dễ dàng kiểm soát
 - Cuối mỗi vòng lặp, phần thi hành được của hệ thống phần mềm được sản sinh theo cách thêm vào dần dần
- Tập trung vào kiến trúc
 - Hệ thống phức tạp được chia thành các mô-đun để dễ dàng triển khai và bảo trì
 - Kiến trúc này được trình bày theo 5 góc nhìn khác nhau

Các nguyên tắc cơ bản (2)

- Dẫn dắt theo các ca sử dụng (use cases)
 - Nhu cầu người dùng thể hiện bởi các ca sử dụng. Các ca sử dụng ảnh hưởng xuyên suốt cho mọi giai đoạn phát triển hệ thống, là cơ sở xác định vòng lặp và tăng trưởng, là căn cứ để phân chia công việc trong nhóm
 - **Nắm bắt nhu cầu:** Phát hiện các ca sử dụng
 - **Phân tích:** Đi sâu vào mô tả các ca sử dụng
 - **Thiết kế và cài đặt:** Xây dựng hệ thống theo các ca sử dụng
 - **Kiểm thử và nghiệm thu hệ thống:** Thực hiện theo các ca sử dụng
- Khống chế các nguy cơ (risks)
 - Phát hiện sớm và loại bỏ các nguy cơ đối với dự án PTPM

Các giai đoạn của RUP (1)

- RUP được tổ chức thành 4 giai đoạn: Khởi đầu (Inception), Chi tiết hóa (Elaboration), Xây dựng (Construction), và Chuyển giao (Transition)



Các giai đoạn của RUP (2)

- Giai đoạn khởi đầu (Inception)

- Cho một cái nhìn tổng quát về hệ thống phần mềm (chức năng, hiệu năng, công nghệ,...) và về dự án PTPM sẽ triển khai (phạm vi, mục tiêu, tính khả thi,...) => Đưa ra kết luận nên phát triển dự án hay loại bỏ?

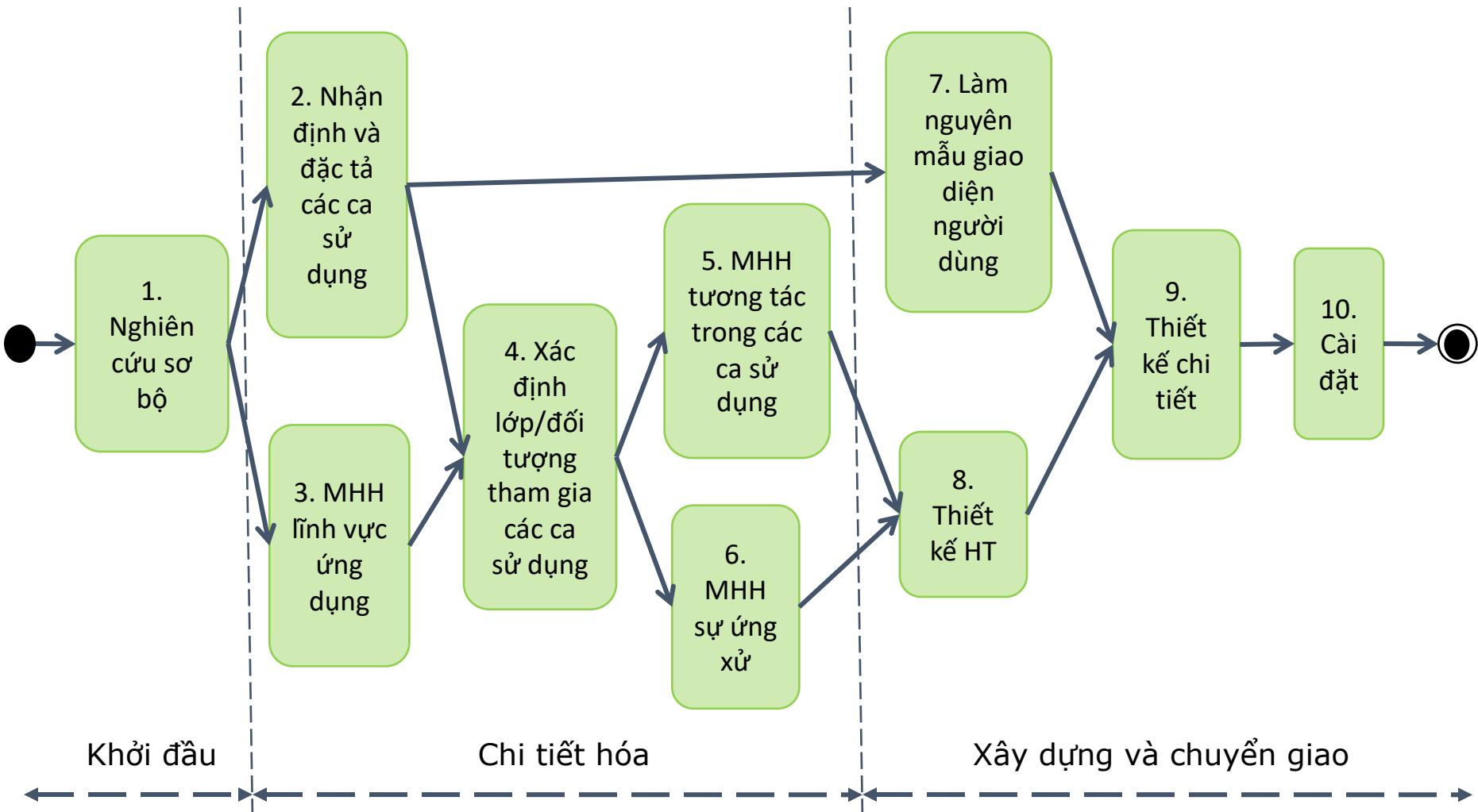
- Giai đoạn chi tiết hóa (Elaboration)

- Phân tích chi tiết hơn về hệ thống (chức năng và cấu trúc tinh)
- Đề xuất một kiến trúc hệ thống (nguyên mẫu)

Các giai đoạn của RUP (3)

- Giai đoạn xây dựng (Construction)
 - Tập trung vào thiết kế và cài đặt hệ thống
 - Kiến trúc hệ thống được chi tiết hóa, chỉnh sửa
 - Kết thúc khi đã cho ra được 1 hệ thống hoàn chỉnh với tài liệu kỹ thuật đi kèm
 - Là giai đoạn tốn nhiều thời gian, công sức nhất
- Giai đoạn chuyển giao (Transition)
 - Chuyển giao hệ thống cho người dùng cuối: chuyển đổi dữ liệu, lắp đặt, kiểm tra, đào tạo,...

Các bước chính của RUP (1)



Các bước chính của RUP (2)

1. Nghiên cứu sơ bộ

- Đưa ra cái nhìn khái quát về hệ thống phần mềm và dự án PTPM
- Đưa ra kết luận: nên/không nên triển khai dự án?

2. Nhận định và đặc tả các ca sử dụng

- Nắm bắt nhu cầu của người dùng, phát hiện các ca sử dụng
- Mỗi ca sử dụng phải được đặc tả (được mô tả) dưới dạng kịch bản và/hoặc một biểu đồ trình tự

3. MHH lĩnh vực ứng dụng

- Đưa ra mô hình biểu đồ lớp phản ánh mọi khái niệm, nghiệp vụ
- Các lớp ở đây là các lớp lĩnh vực (không phải là các lớp đối tượng)

Các bước chính của RUP (3)

4. Xác định các đối tượng/lớp tham gia ca sử dụng

- Với mỗi ca sử dụng, phát hiện các lớp lĩnh vực, lớp điều khiển, lớp biên

5. MHH tương tác trong các ca sử dụng

- Các đối tượng tương tác bằng cách trao đổi thông điệp
- Tạo kịch bản của ca sử dụng: biểu đồ trình tự, biểu đồ giao tiếp

6. MHH sự ứng xử

- Các đối tượng điều khiển có khả năng ứng xử đối với các sự kiện đến từ bên ngoài để điều khiển
- Sử dụng biểu đồ trạng thái để mô tả hành vi ứng xử của các đối tượng điều khiển

Các bước chính của RUP (4)

7. Làm nguyên mẫu giao diện người dùng

- Sử dụng các bộ tạo lập giao diện người dùng (graphical user interface – GUI) để làm sớm nguyên mẫu giao diện, giúp cho việc mô hình hóa và cài đặt hệ thống dễ dàng hơn

8. Thiết kế hệ thống

- Thiết kế kiến trúc tổng thể của hệ thống
- Chia thành các hệ thống con, chọn lựa loại hình điều khiển thích hợp
- Dùng biểu đồ thành phần mô tả các thành phần vật lý
- Dùng biểu đồ triển khai mô tả cách bố trí, triển khai các thành phần thực thi của hệ thống vào các phần cứng và nền tảng hạ tầng
- Kiến trúc khách/chủ (client/server) là một kiến trúc hệ thống hay được sử dụng

Các bước chính của RUP (5)

9. Thiết kế chi tiết

- Thiết kế các lớp, các liên kết, các thuộc tính, các phương thức
- Xác định các giải pháp cài đặt hệ thống

10. Cài đặt

- Lập trình và kiểm thử
- Hệ thống được nghiệm thu dựa theo các ca sử dụng



TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Thank you for your attention!
Q&A

