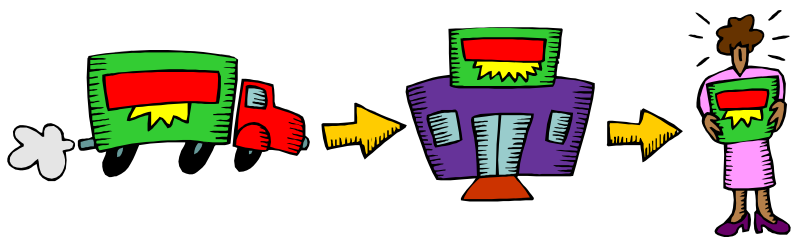
Công nghệ phần mềm

Tiến trình và mô hình tiến trình phần mềm



Nội dung

- Định nghĩa tiến trình và mô hình tiến trình
- Lặp tiến trình
- Mô tả các mô hình tiến trình
  - Thác nước, tiến hóa, dựa trên thành phần
  - Lựa chọn các mô hình
- Các hoạt động chung nhất của mọi tiến trình
- Đối phó, kiểm soát sự thay đổi yêu cầu phần mềm

Các khái niệm

- Tiến trình (software process): là tập hợp các hoạt động để tạo ra phần mềm
  - Đặc tả - Specification;
  - Thiết kế và cài đặt – Design and Implementation;
  - Kiểm định - Validation;
  - Cải tiến - Evolution.
- Mô hình tiến trình (process model) là thể hiện trừu tượng của tiến trình.
- Lặp tiến trình (iteration): lặp lại một tiến trình để đạt tới mục tiêu mong đợi
  - Kết quả của một lần lặp được sử dụng như điểm bắt đầu của lần lặp tiếp theo.

Các mô hình tiến trình

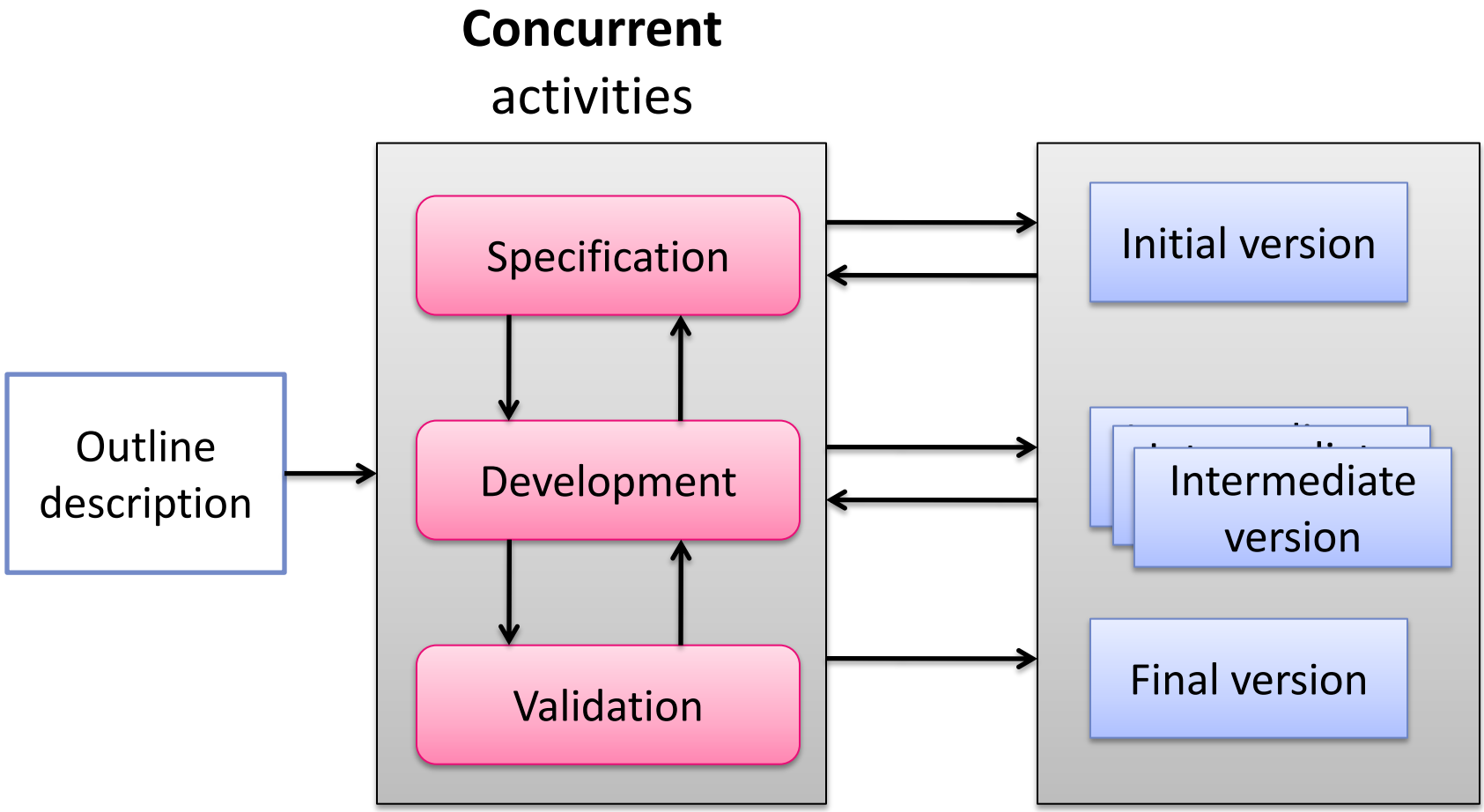
- Mô hình thác nước – Waterfall
  - Phân tách các giai đoạn đặc tả và phát triển.
- Tiến hóa - Evolutionary development
  - Đặc tả, phát triển và kiểm định được thực hiện xen kẽ nhau.
- Component-based software engineering
  - Hệ thống được kết hợp từ các thành phần có sẵn.
- Có nhiều biến thể của các mô hình

Waterfall model

Ưu, nhược điểm của mô hình thác nước

- **Khó đáp ứng khi khách hàng thay đổi yêu cầu.**
  - Ít hệ thống có yêu cầu cố định.
- Chậm có phiên bản thực hiện được
  - Sai sót phát hiện muộn gây tổn thất lớn
- Chỉ phù hợp khi yêu cầu được hiểu rõ
- Bảo trì thuận lợi
  - Tài liệu được làm tốt

Evolutionary development



Evolutionary development

- Phát triển thăm dò - Exploratory development
  - Làm việc với khách hàng và cải tiến hệ thống từng bước
    - Bắt đầu với những yêu cầu chưa đầy đủ
    - Thêm những tính năng mới như khách hàng yêu cầu
    - Lặp lại các bước cho đến khi có phiên bản cuối cùng
- Bản mẫu - Throw-away prototyping
  - Mục tiêu là để hiểu yêu cầu khách hàng

Evolutionary development

- Hạn chế
  - Thiếu trực quan
  - Hệ thống thường có cấu trúc nghèo nàn
  - Đòi hỏi kỹ năng đặc tả
    - Eg. Rapid prototyping language
- Khả năng ứng dụng
  - Hệ thống có nhiều tương tác, nhỏ và vừa
  - Cho các phần của hệ lớn (e.g. the user interface)
  - Cho các hệ có vòng đời ngắn

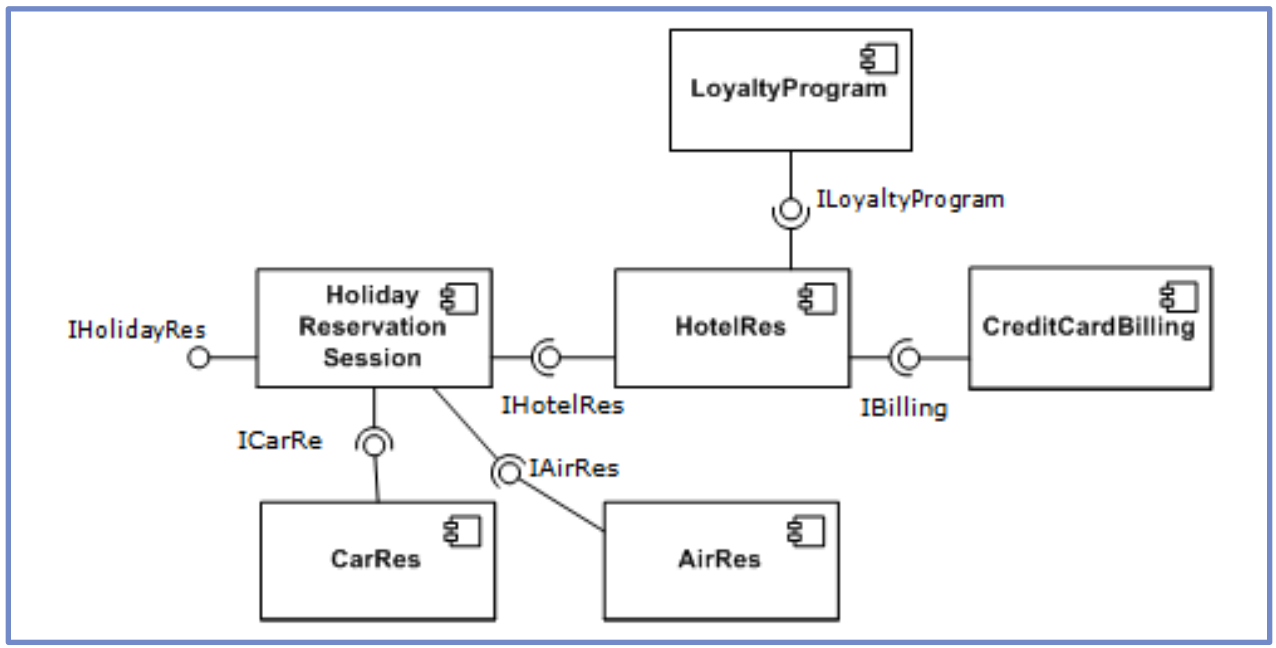
Component-based software engineering

Thành phần phần mềm (software components)

- **Thành phần là một đơn vị phần mềm (gói – package hoặc mô đun – module) đóng gói một tập các dịch vụ có liên quan đến nhau.**
- **Mỗi thành phần đều có tên (name), giao diện (interface) và mã (code).**

Component-based software engineering

- Component-based software: hệ thống được kết hợp từ các thành phần có sẵn

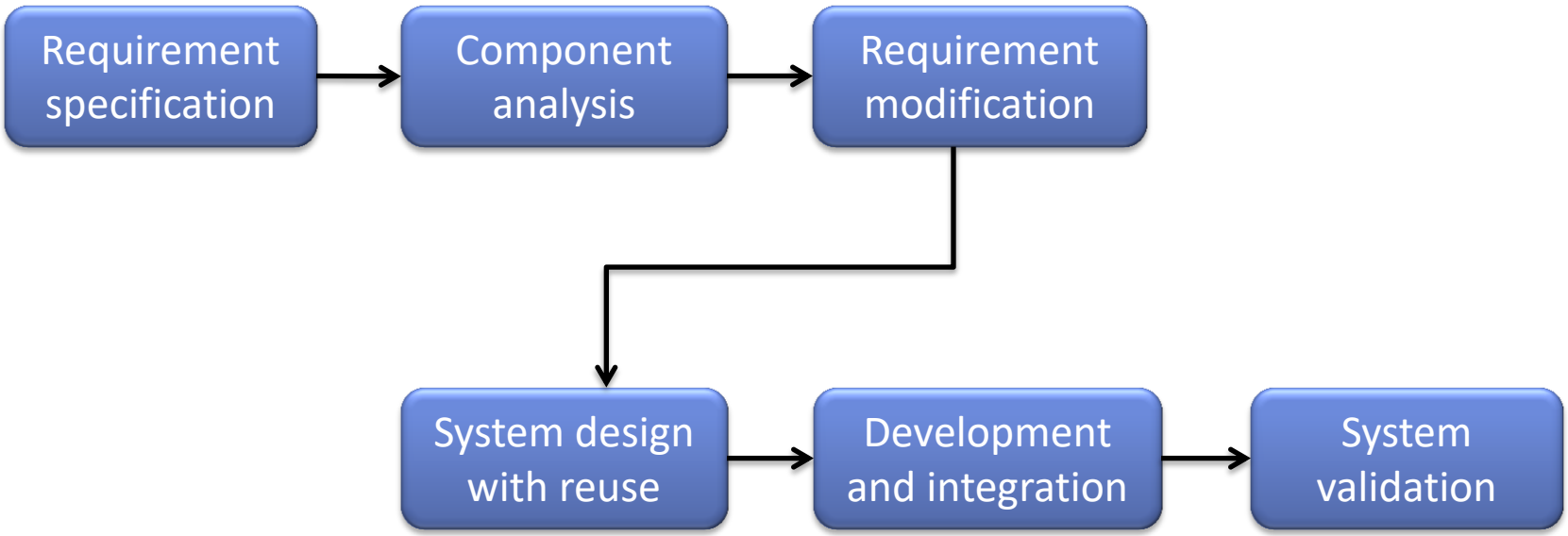


  - Reusability
  - Replacing component

Component-based software engineering

- Dựa trên việc sử dụng lại một cách có hệ thống
  - Hệ thống được tích hợp từ những thành phần có sẵn.
- Các bước
  - Đặc tả yêu cầu
  - Phân tích thành phần
  - Sửa đổi yêu cầu
  - Thiết kế hệ thống với sử dụng lại
  - Phát triển và tích hợp
  - Thẩm định sản phẩm

Reuse-oriented development



Các hoạt động chung nhất của mọi tiến trình

1. Đặc tả yêu cầu – Specification
2. Thiết kế và cài đặt - Design and implementation
3. Thẩm định - Validation
4. Tiến hóa - Evolution

Các hoạt động chung nhất của mọi tiến trình

1. Đặc tả yêu cầu – Specification
   - Định nghĩa các dịch vụ và các ràng buộc cho phần mềm.
2. Thiết kế và cài đặt - Design and implementation
3. Thẩm định - Validation
4. Tiến hóa - Evolution

Các hoạt động chung nhất của mọi tiến trình

1. Đặc tả yêu cầu – Specification
2. Thiết kế và cài đặt - Design and implementation
   - Thiết kế cấu trúc phần mềm
   - Chuyển cấu trúc phần mềm thành chương trình thực thi được.
3. Thẩm định - Validation
4. Tiến hóa - Evolution

Các hoạt động chung nhất của mọi tiến trình

1. Đặc tả yêu cầu – Specification
2. Thiết kế và cài đặt - Design and implementation
3. Thẩm định – Validation
    - Chỉ ra rằng hệ thống thỏa mãn yêu cầu khách hàng
4. Tiến hóa - Evolution

Các hoạt động chung nhất của mọi tiến trình

1. Đặc tả yêu cầu – Specification
2. Thiết kế và cài đặt - Design and implementation
3. Thẩm định – Validation
4. Tiến hóa – Evolution
    - Làm thích nghi hệ thống với nghiệp vụ mới, môi trường vận hành mới, …

Đối phó với sự thay đổi yêu cầu phần mềm

- Thay đổi yêu cầu là chắc chắn xảy ra với mọi dự án phần mềm
  - Tốn chi phí cho việc làm lại các phần liên quan (cost of rework)
- Các cách tiếp cận để giảm chi phí cho việc làm lại:
  - Tránh sự thay đổi
  - Chấp nhận thay đổi
  - Mô hình phát triển xoắn ốc

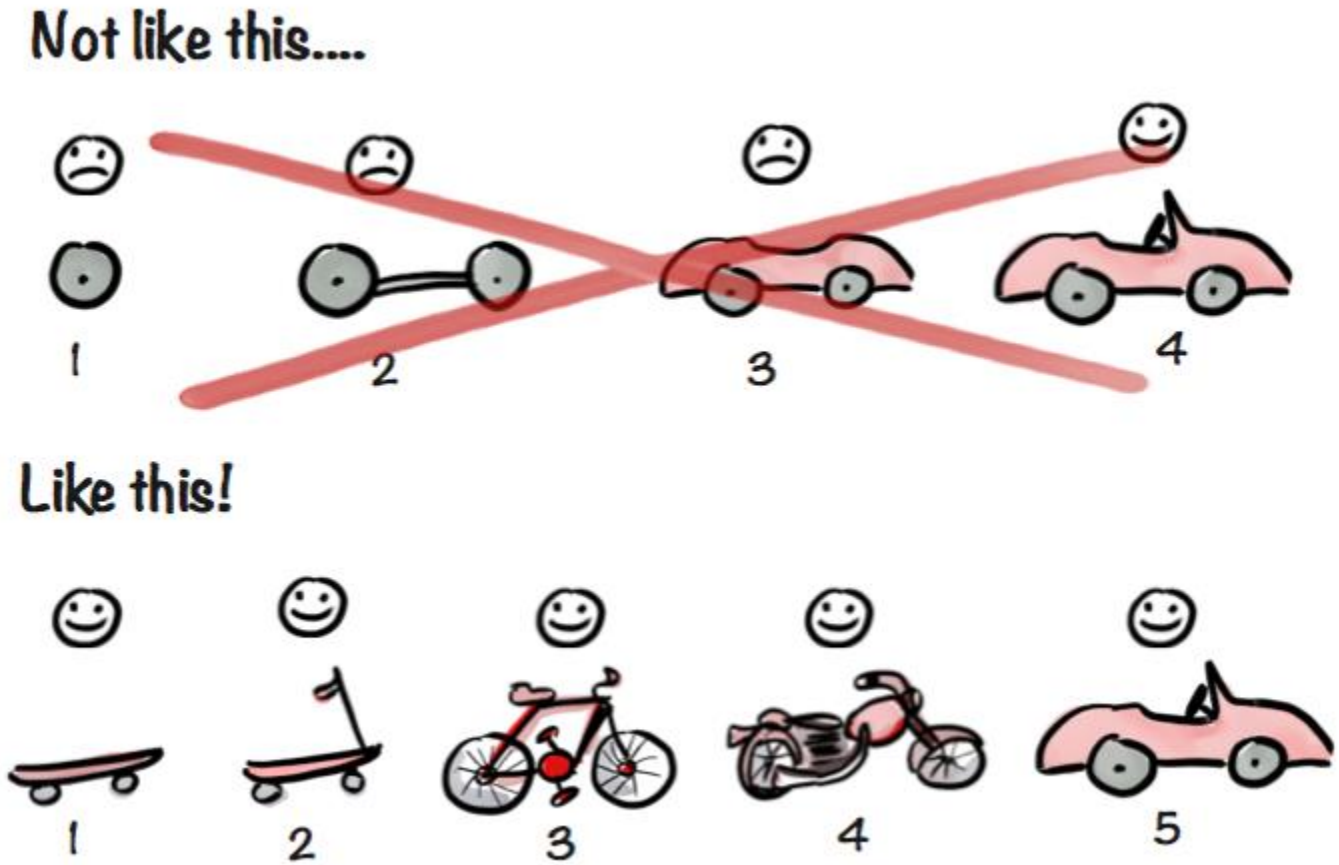
Tránh sự thay đổi

- Làm bản mẫu để xác nhận và làm mịn yêu cầu.
  - Throw away prototype
  - Phần mềm vận hành được

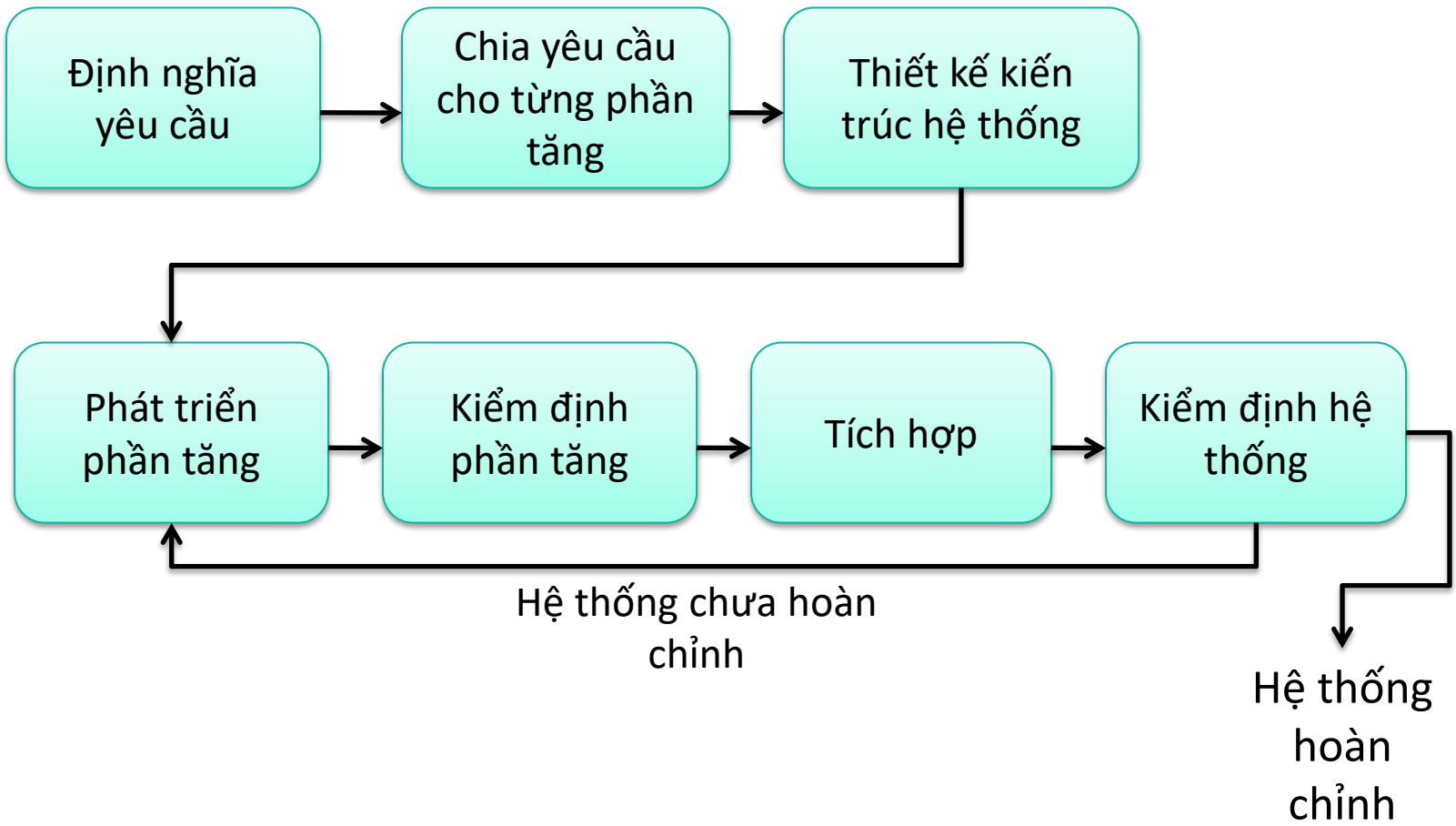
Chấp nhận thay đổi

- Chuyển giao dần từng phần của hệ thống - Incremental delivery
  - Sản phẩm chia thành từng phần tăng (phiên bản ở mỗi lần lặp tiến trình) theo yêu cầu chức năng
  - Yêu cầu người dùng được xếp ưu tiên theo thứ tự phần tăng

Incremental delivery



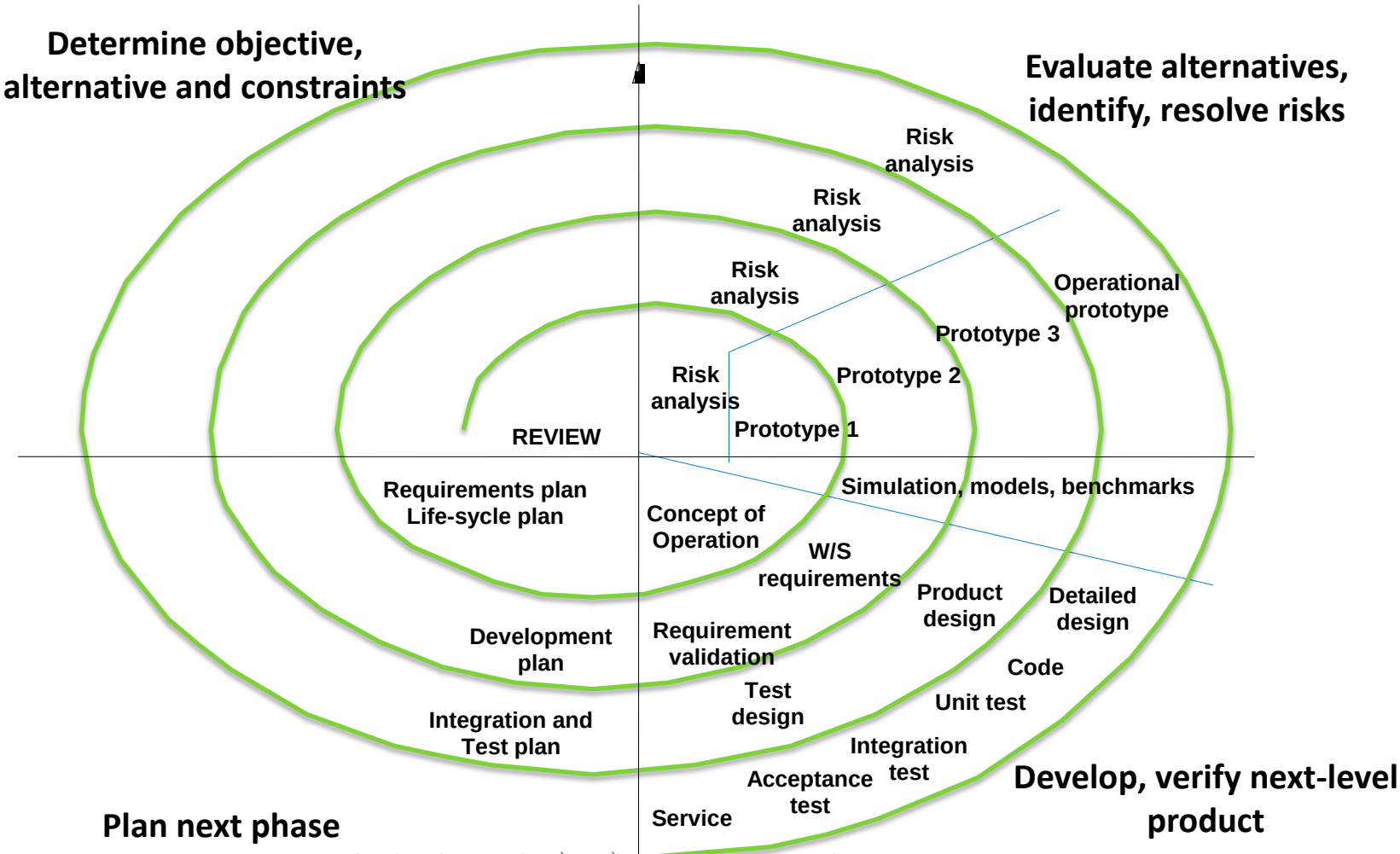
Incremental development



Ưu điểm

- Người dùng có sản phẩm dùng sớm
- Các phiên bản sớm đóng vai trò như bản mẫu để giúp thu thập yêu cầu mới cho phiên bản kế tiếp.
- Các yêu cầu có độ ưu tiên cao được kiểm thử nhiều lần.

Mô hình phát triển xoắn ốc
Spiral model of the software process



Spiral development

- Mỗi vòng lặp thể hiện một pha của tiến trình phát triển phần mềm
  - Nghiên cứu khả thi
  - Yêu cầu
  - Thiết kế
  - Phát triển ...
- Kết hợp cả tránh sự thay đổi và chấp nhận sự thay đổi
- Rủi ro được đánh giá và được giải quyết ở mỗi lần lặp

Tổng kết

- Tiến trình - process
- Mô hình tiến trình – process models
- Lặp tiến trình – process iteration
- Mô hình thác nước
- Mô hình tiến hóa
- Mô hình phát triển dựa trên thành phần
- Các hoạt động chung nhất của mọi tiến trình
- Đối phó, kiểm soát sự thay đổi yêu cầu phần mềm