Viện Đại học Mở Hà Nội
Khoa Công nghệ Điện tử - Thông tin

Lập trình ứng dụng

Quy Trình Sản Xuất Phần Mềm

Các nội dung chính

- Giới thiệu chung
- Các khái niệm cơ bản
- Các loại phần mềm
- Giới thiệu các loại quy trình phát triển phần mềm phổ biến

Giới thiệu chung

- Phần mềm có đặc điểm là **trừu tượng** và **không chạm đến được** (intangible). Điều này làm cho phần mềm rất dễ trở nên phức tạp và khó hiểu;
- Việc phát triển phần mềm cần theo các *quy trình sản xuất* khoa học để đảm bảo hai điều kiện:
  - Chất lượng phần mềm tốt: thỏa mãn các nhu cầu người dùng;
  - Tỉ lệ *Giá thành/Chi phí phát triển* cao nhất.

Các khái niệm cơ bản

- **Phần mềm** (sản phẩm phần mềm), bao gồm:
  - **Chương trình** (Program): là phần được thi hành trên máy tính;
  - **Dữ liệu** (Data): gồm các cấu trúc dữ liệu, cơ sở dữ liệu lưu giữ các dữ liệu vào và ra của chương trình;
  - **Tài liệu** (Documentation): tài liệu hệ thống, tài liệu người dùng.

Các khái niệm cơ bản

- **Kỹ thuật phần mềm** (Software Engineering): Là một chuyên ngành kỹ thuật mà quan tâm đến tất cả các khía cạnh của việc sản xuất phần mềm, với mục tiên sản xuất ra các **sản phẩm phần mềm** *đa dạng*, *chất lượng cao*, một cách *hiệu quả nhất*.
- Cụ thể hơn, KTPM cần tạo ra các *Quy trình Sản xuất Phần mềm* (hay *Mô hình*) hiệu quả và khả thi.

Quy trình Sản xuất Phần mềm

- Là một dãy các giai đoạn và các hoạt động trong đó, cũng như các kết quả kèm theo. Kết quả cuối cùng chính là phần mềm cần phải xây dựng, đáp ứng được các yêu cầu của người dùng, và hoàn thành theo đúng kế hoạch về thời gian và ngân sách;
- Có ba giai đoạn chính trong tiến trình phần mềm:
  - Giai đoạn định nghĩa (definition phase),
  - Giai đoạn phát triển (development phase),
  - Giai đoạn hỗ trợ (support phase).

Quy trình Sản xuất Phần mềm

- **Giai đoạn định nghĩa**: tập trung vào làm rõ ***Cái gì***, bao gồm:
  - Thông tin gì cần xử lý, bao gồm thông tin đầu vào và đầu ra.
  - Các chức năng gì cần thực hiện.
  - Hành vi nào của hệ thống sẽ được mong đợi.
  - Các tiêu chuẩn hợp lệ nào để đánh giá được sự đúng đắn và thành công của hệ thống.

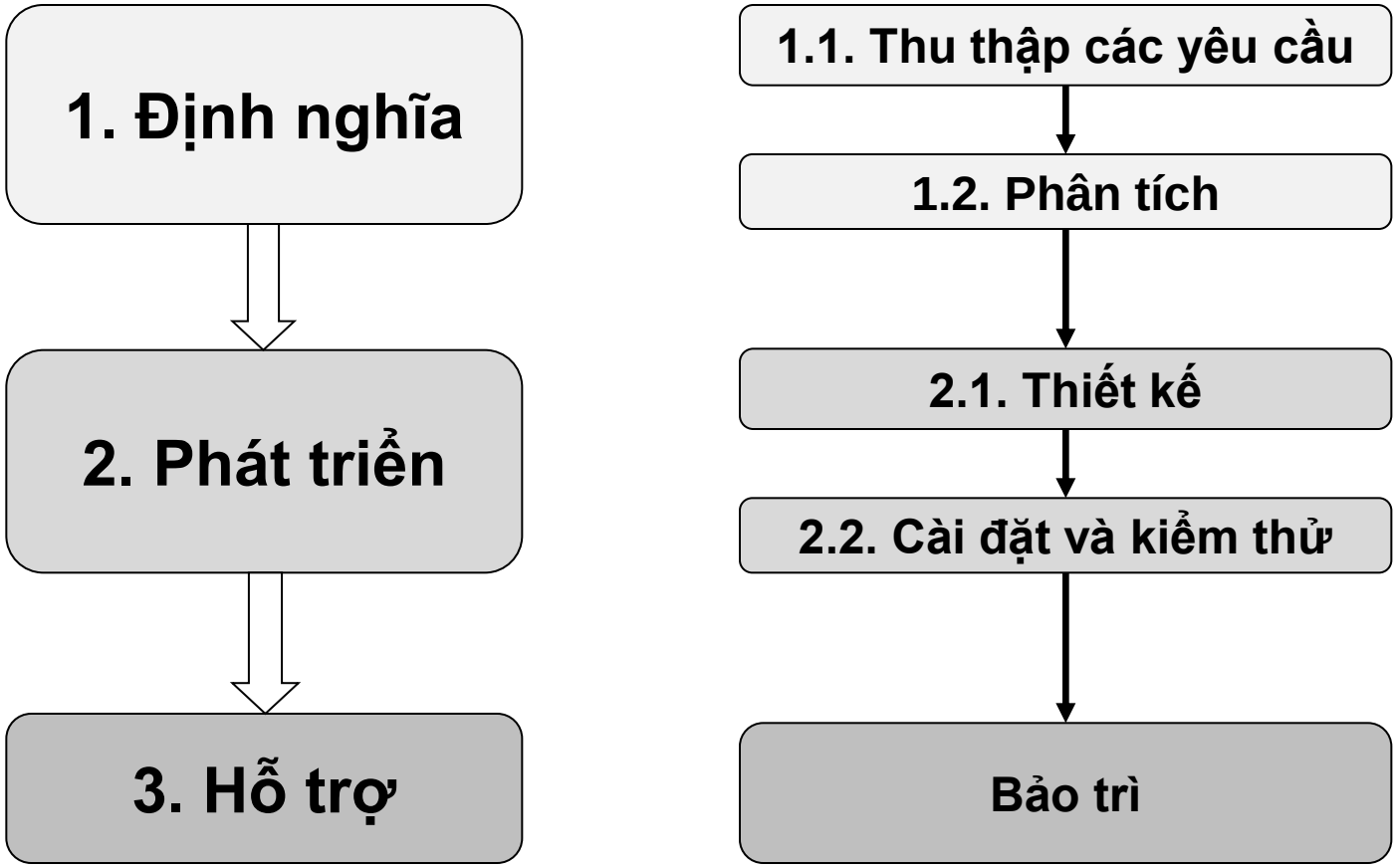
Quy trình Sản xuất Phần mềm

- **Giai đoạn phát triển**: tập trung vào ***Làm thế nào***, bao gồm:
  - Kiến trúc hệ thống (system architecture) được tổ chức thế nào.
  - Các chức năng được cài đặt và liên kết với nhau thế nào.
  - Tổ chức các cấu trúc dữ liệu, cơ sở dữ liệu thế nào.
  - Chuyển từ thiết kế sang cài đặt thế nào?
  - Việc kiểm thử sẽ được thực hiện thế nào?

Quy trình Sản xuất Phần mềm

- **Giai đoạn hỗ trợ**: còn gọi là giai đoạn bảo trì, tập trung vào việc ứng phó với các thay đổi của hệ thống phần mềm, bao gồm:
  - Sửa lỗi (Correction)
  - Làm thích ứng (Adaptation)
  - Nâng cấp (Upgrade)

Quy trình Sản xuất Phần mềm



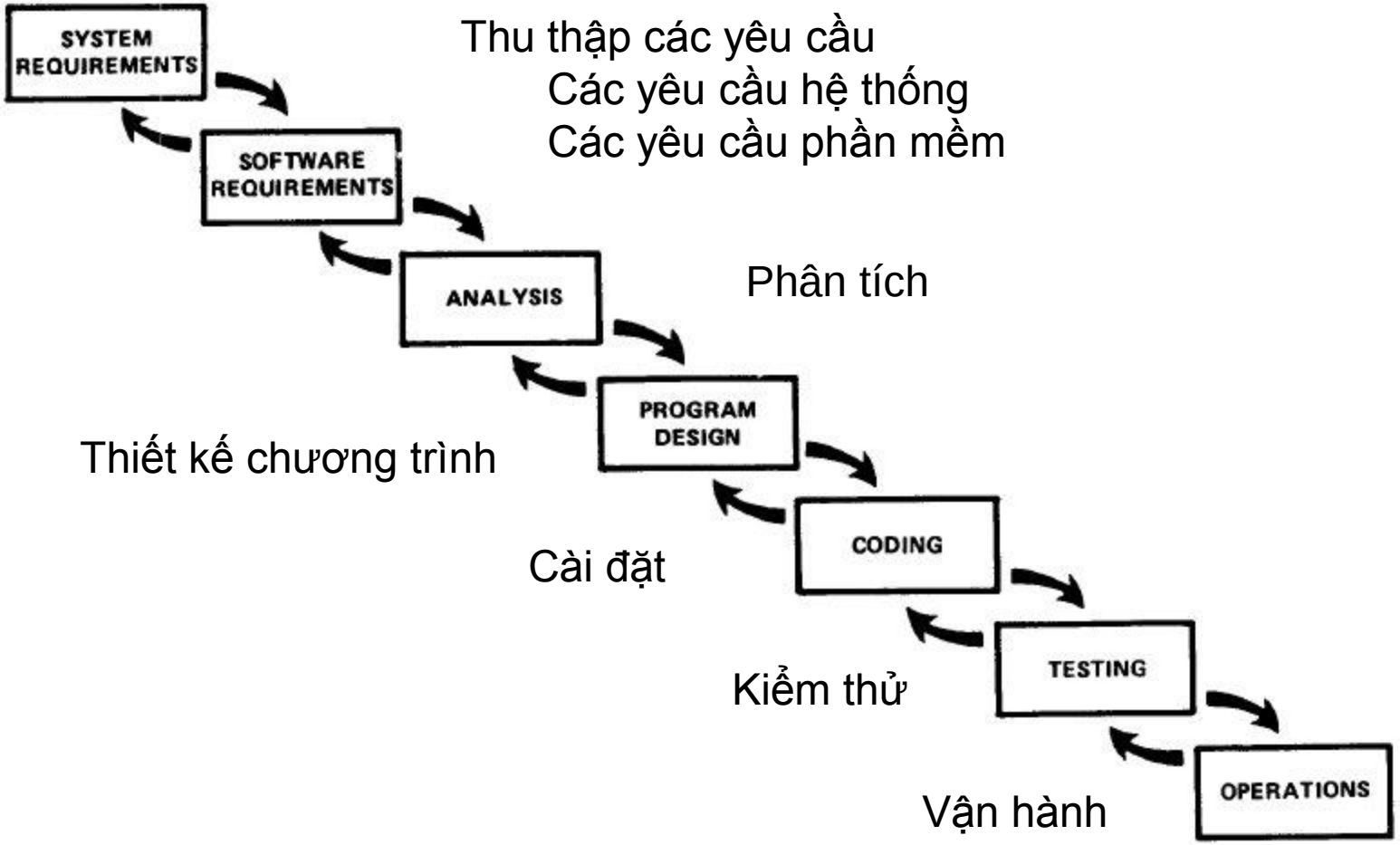
Các loại phần mềm

- Phần mềm hệ thống (system software)
- Phần mềm thời gian thực (real time sw)
- **Phần mềm quản lý** (business sw): cũng được gọi là ***hệ thông tin quản lý*** (management information system – MIS)
- Phần mềm khoa học và công nghệ (engineering and scientific sw)
- Phần mềm nhúng (embedded sw)
- Phần mềm văn phòng (office sw)
- Phần mềm Web (Web-based sw)
- Phần mềm trí tuệ nhân tạo (artificial intelligence sw)
- V.v.

Các Quy Trình Phần Mềm phổ biến

- Tuyến tính cổ điển (mô hình thác nước – Waterfall)
- Bản mẫu (Prototyping)
- Phát triển nhanh (RAD - Rapid Application Development)
- Tăng trưởng (Incremental model)
- Xoáy ốc (Spiral model)
- Hướng đối tượng (Object-Oriented model)

Mô hình tuyến tính cổ điển*



Thu thập các yêu cầu
Các yêu cầu hệ thống
Các yêu cầu phần mềm

Phân tích

Thiết kế chương trình

Cài đặt

Kiểm thử

Vận hành

Mô hình tuyến tính cổ điển

- Mô hình này có một số đặc điểm như sau:
  - Các bước được tiến hành tuần tự, kết thúc bước trước thì mới thực hiện đến bước sau
  - Thời gian thực hiện mỗi bước thường kéo dài do phải làm thật hoàn chỉnh
  - Thường chỉ tiếp xúc với người dùng vào giai đoạn đầu và giai đoạn cuối. Người dùng thường không tham gia vào các bước ở giữa, như từ thiết kế, cài đặt và đến tích hợp

Mô hình tuyến tính cổ điển

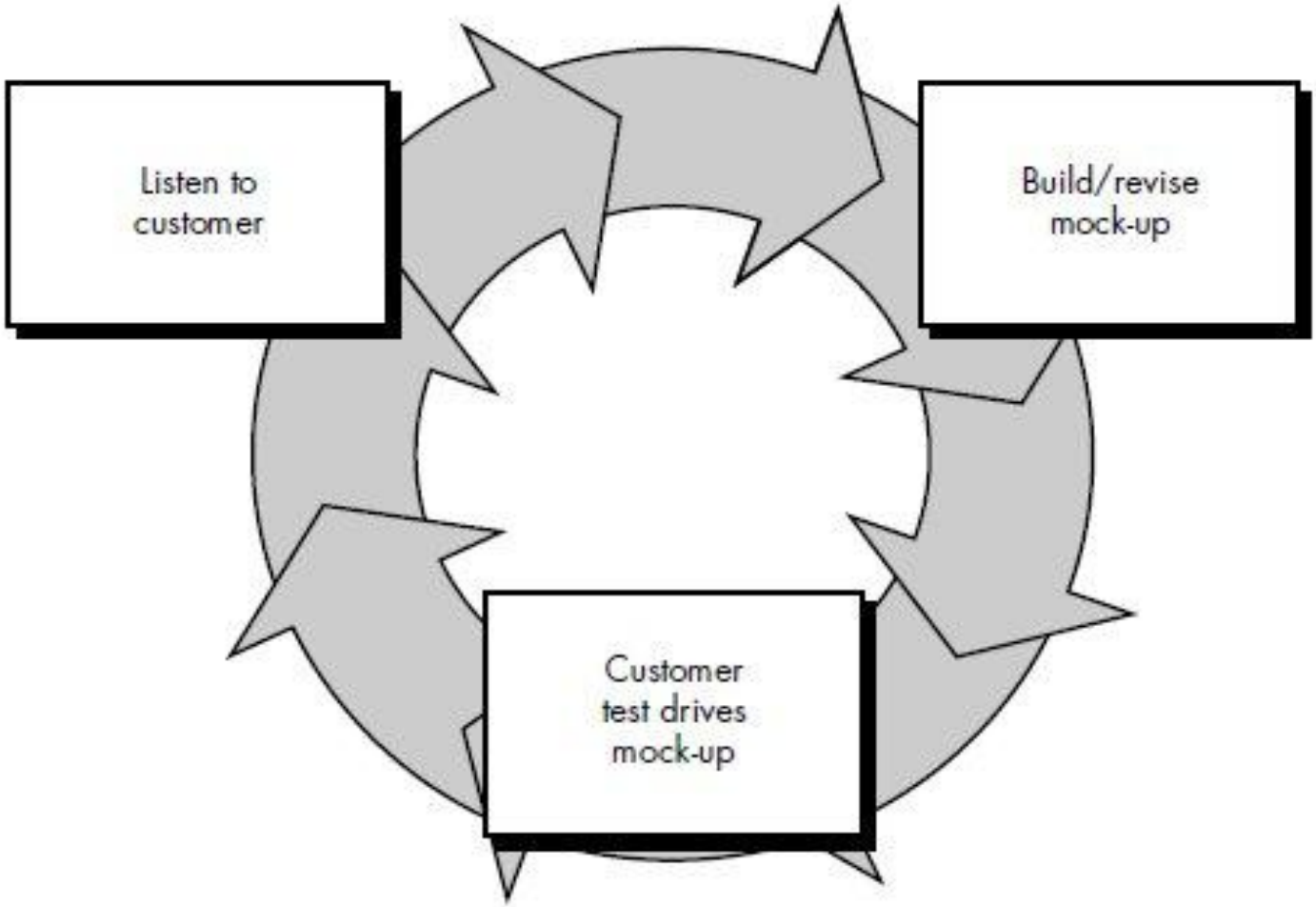
- **Ưu điểm**:
  - Đơn giản và rõ ràng;
  - Đóng vai trò như một mẫu tham chiếu cho các mô hình khác;
  - Vẫn còn được sử dụng rộng rãi cho đến nay.
- **Nhược điểm**:
  - Không dễ dàng cho việc thu thập đầy đủ và tường minh tất cả các yêu cầu hệ thống ngay từ ban đầu;
  - Người dùng phải chờ đến cuối cùng mới có được hệ thống để dùng, nên thời gian chờ đợi là khá lâu, có khi đến hàng năm. Khi đó có thể có các yêu cầu mới đã phát sinh, dễ dẫn khả năng hệ thống không còn đáp ứng được kỳ vọng của người dùng;
  - Dễ dẫn đến tình trạng "**blocking states**", tức là khi có một nhóm bị chậm tiến độ, thì các nhóm khác phải chờ, và thời gian chờ đợi thậm chí vượt quá thời gian làm việc.

Mô hình bản mẫu

- Thông thường trong thực tế, các yêu cầu của hệ thống khó có thể xác định rõ ràng và chi tiết ngay trong gia đoạn đầu của dự án phần mềm vì:
  - **Người dùng** cũng chỉ đưa ra các mục tiêu tổng quát của phần mềm, chứ cũng chưa định rõ được một cách chi tiết các chức năng cụ thể, hay các thông tin chi tiết đầu vào, đầu ra như thế nào;
  - **Nhà phát triển** cũng chưa xác định rõ ràng ngay các yêu cầu, cũng như chắc chắn về chất lượng phần mềm, cũng như khả năng thỏa mãn của khách hàng.

  → **mô hình bản mẫu**

Mô hình bản mẫu



Mô hình bản mẫu

Gồm các giai đoạn:

- **Thu thập các yêu cầu** (requirements gathering): khách hàng và nhà phát triển sẽ gặp nhau để xác định ra các mục tiêu tổng thể của phần mềm. Sau đó họ sẽ định ra phần nào đã rõ, phần nào cần phải định nghĩa thêm.
- **Thiết kế nhanh (quick design):** thiết kế này tập trung vào những phần mà khách hàng có thể nhìn thấy được (giao diện, các dữ liệu vào, ra). Sau đó, từ thiết kế này, một bản mẫu sẽ được xây dựng.
- **Kiểm tra và đánh giá bản mẫu:** Bản mẫu này sẽ được dùng để cho phép người dùng đánh giá, nhằm làm rõ hơn các yêu cầu của họ. Đồng thời, thông qua bản mẫu, người phát triển hệ thống cũng hình dung cụ thể hơn về những yêu cầu của khách hàng, cũng như khả năng cài đặt và hiệu quả hoạt động của hệ thống.

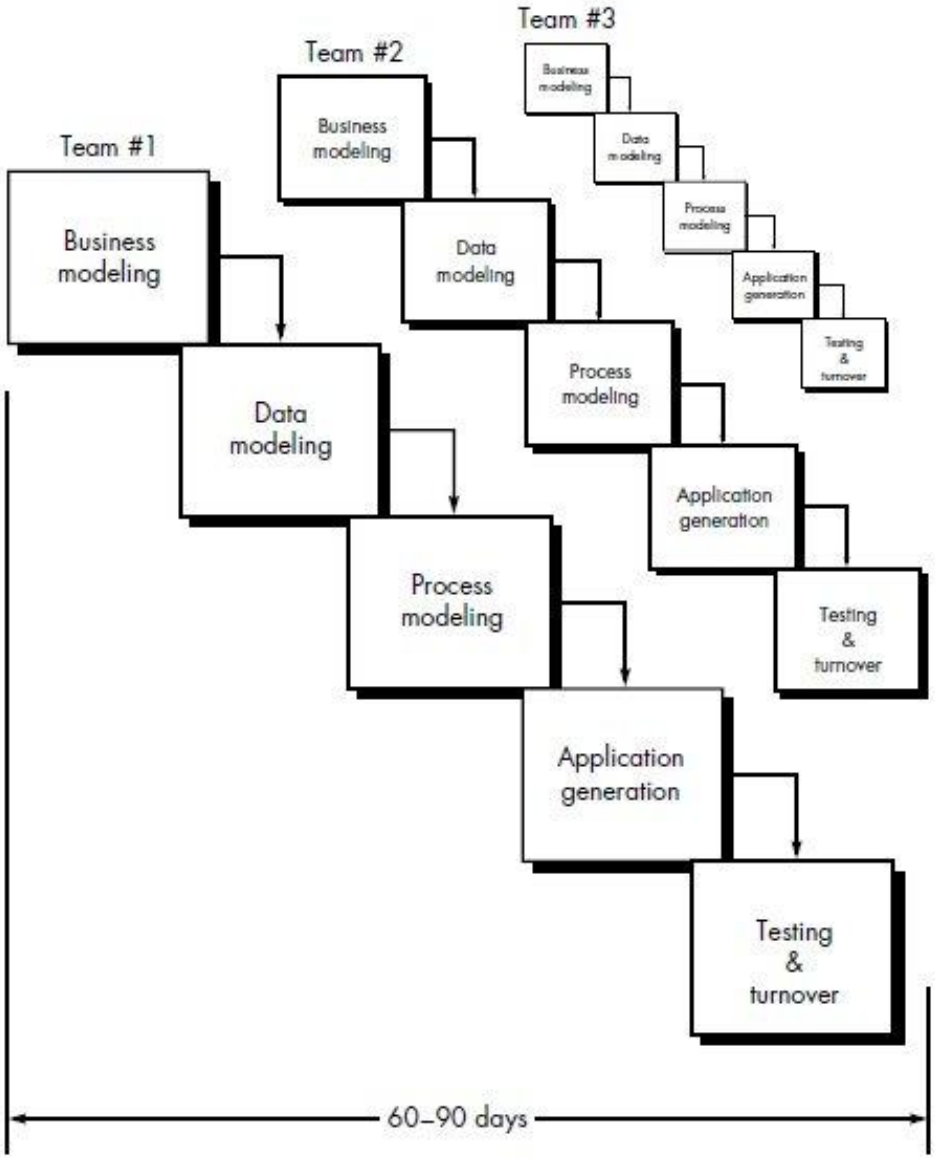
Mô hình bản mẫu

- **Ưu điểm**:
  - Cho phép người dùng xác định yêu cầu của mình rõ ràng và cụ thể hơn, đồng thời nhà phát triển cũng nắm được chính xác hơn các yêu cầu đó.
  - Cả người dùng và nhà phát triển thường đều thích mô hình này, do người dùng luôn cảm nhận được hệ thống thực sẽ như thế nào, và nhà phát triển cũng luôn có cái để xây dựng và dần hoàn thiện.
- **Nhược điểm**:
  - Để có được bản mẫu nhanh, việc thiết kế cũng được làm nhanh, nên thường được làm không cẩn thận. Điều này dễ dẫn đến các thiết kế có tính chắp vá, không có cái nhìn tổng thể và dài hạn.
  - Việc làm bản mẫu nhanh cũng thường kéo theo việc lựa chọn các công cụ cài đặt vội vàng, không cẩn thận, (như ngôn ngữ lập trình, hệ quản trị cơ sở dữ liệu,v.v). Điều này sẽ ảnh hưởng đến các giai đoạn phát triển sau khi quy mô và yêu cầu của hệ thống ngày càng lớn lên.

Mô hình RAD

- Là mô hình tiến trình phát triển phần mềm tăng trưởng, nhưng nhấn mạnh vào chu trình phát triển phần mềm có thời gian rất ngắn. Mô hình này gồm các giai đoạn:
  - **Mô hình hóa nghiệp vụ** (Business modeling): mô hình hóa các luồng thông tin nghiệp vụ giữa các chức năng nghiệp vụ
  - **Mô hình hóa dữ liệu** (Data modeling): từ các thông tin nghiệp vụ, các thực thể dữ liệu, các thuộc tính của chúng, và các liên kết giữa các thực thể này sẽ được xác định và được mô hình hóa.
  - **Mô hình hóa xử lý** (Process modeling): Mô tả các chức năng xử lý trên các đối tượng dữ liệu đã được xác định ở giai đoạn trên.
  - **Sản sinh ứng dụng** (Application generation): RAD sử dụng các kỹ thuật công nghệ phần mềm thế hệ thứ 4, cho phép dễ dàng sản sinh mã chương trình từ các đặc tả và thiết kế trừu tượng. Các kỹ thuật này cũng cho phép tái sử dụng các thành phần chương trình có sẵn (kết hợp mô hình Component-based development).
  - **Kiểm thử và bàn giao** (Testing and turnover): phần ứng dụng đã xây dựng sẽ được kiểm tra và bàn giao cho bên tích hợp hệ thống.

Mô hình RAD


Team #1
Business modeling
Data modeling
Process modeling
Application generation
Testing & turnover
Team #2
Business modeling
Data modeling
Process modeling
Application generation
Testing & turnover
Team #3
Business modeling
Data modeling
Process modeling
Application generation
Testing & turnover
60–90 days

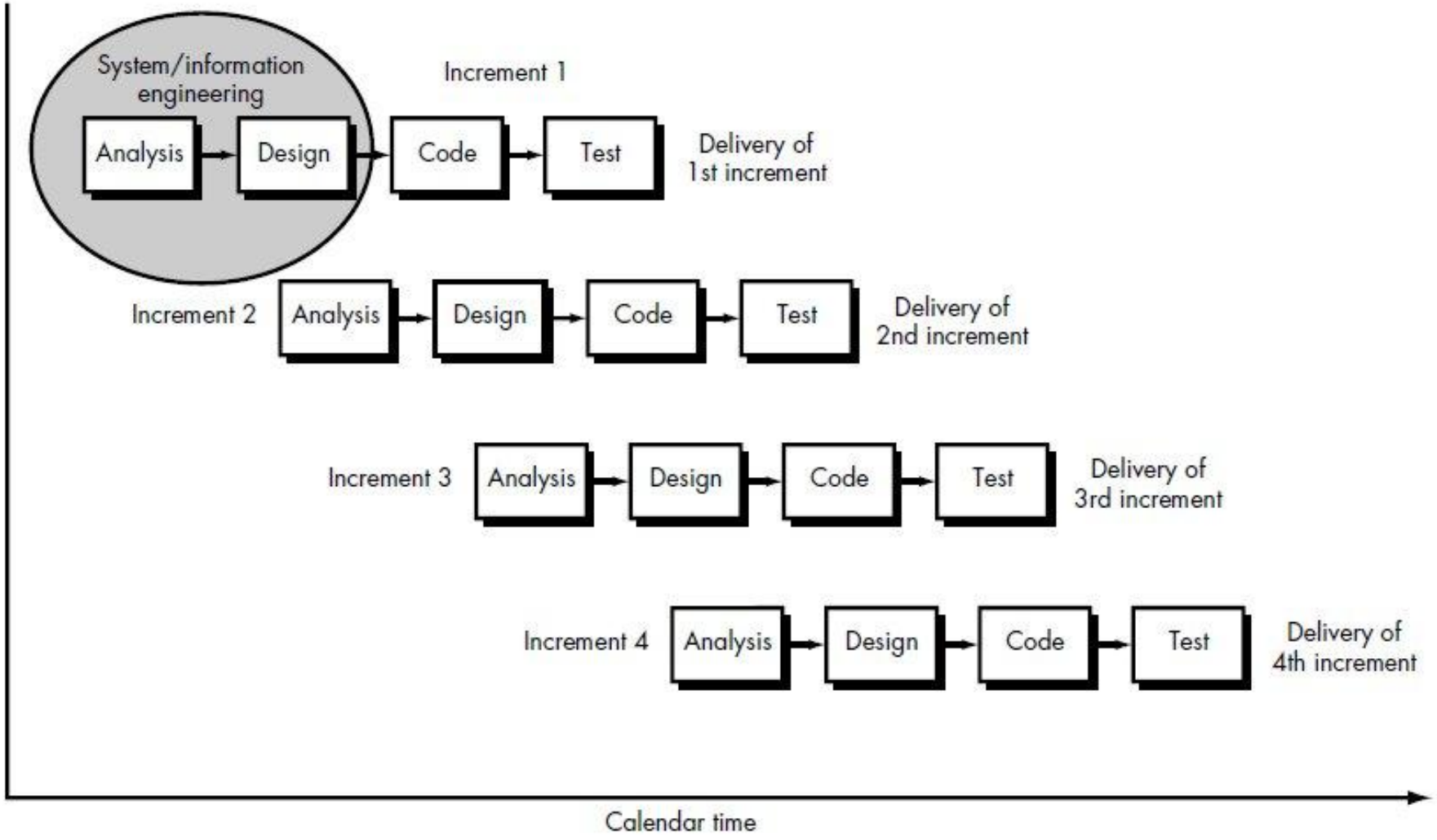
Mô hình RAD

- **Ưu điểm**:
  - Tận dụng các công nghệ mới trong phát triển hệ thống, cho phép hoàn thành hệ thống trong thời gian ngắn hơn đáng kể.
  - Khuyến khích việc tái sử dụng các thành phần của chương trình
- **Nhược điểm:**
  - Không phù hợp với các phần mềm mà không có sự phân chia modul rõ ràng,
  - Đòi hỏi tài nguyên và chi phí phát triển cao như số lượng nhân lực nhiều, công cụ CASE thế hệ 4 đắt tiền

Mô hình tăng trưởng

- Là sự kết hợp của mô hình tuyến tính và triết lý lặp lại của mô hình bản mẫu.
- Phần mềm được chia thành các ***phần tăng trưởng*** (increment), trong đó mỗi phần là một sản phẩm hoàn chỉnh (đã chạy được và có thể bàn giao cho người dùng). Đồng thời *phần tăng trưởng* sau sẽ bổ sung thêm tính năng còn thiếu trong những phần trước.

Mô hình tăng trưởng



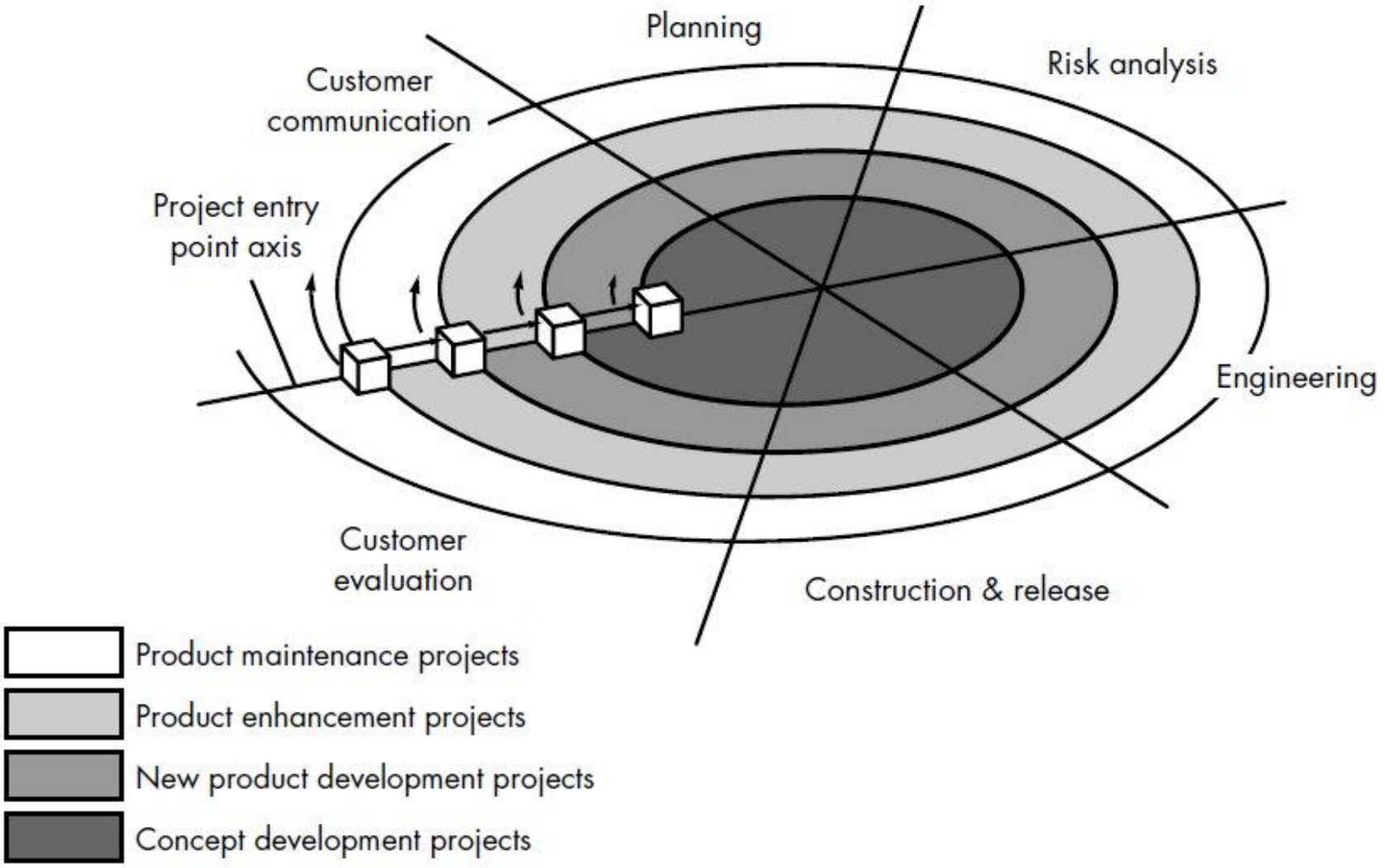
Mô hình tăng trưởng

- **Ưu điểm**
  - Kết hợp được các ưu điểm của các mô hình tuyến tính và làm bản mẫu;
  - Rất phù hợp khi số lượng nhân viên hạn chế, và người dùng có đòi hỏi phải sớm có hệ thống thử nghiệm.
- **Nhược điểm**
  - Việc gấp gáp đưa ra các thành phần tăng trưởng cũng có thể gây ra sự manh mún trong phân tích và thiết kế;
  - Khó khăn trong việc đảm bảo tính tương thích (compatibility) giữa các thành phần tăng trưởng.

Mô hình xoáy ốc

- Cũng là một mô hình tiến hóa kết hợp đặc tính lặp lại của mô hình bản mẫu và tính hệ thống của mô hình thác nước cổ điển;
- Mô hình này cũng cho phép tạo ra một dãy các *phiên bản tăng trưởng* (incremental release). Tuy nhiên khác với mô hình tăng trưởng, các phiên bản đầu tiên của mô hình xoáy ốc thường chỉ là các mô hình trên giấy hoặc bản mẫu (prototype). Đến các phiên bản sau thì mới là các bản chạy được và càng ngày càng hoàn chỉnh.

Mô hình xoáy ốc



Mô hình xoáy ốc

- Mô hình này phân chia thành các giai đoạn, được gọi là các ***vùng nhiệm vụ*** (task regions).
- Số lượng vùng nhiệm vụ có thể thay đổi, và thường có từ 3 cho đến 6 vùng.
- Mỗi vùng lại bao gồm ***một tập các nhiệm vụ*** (set of tasks), và số lượng cũng thay đổi tùy theo tính chất của dự án.

Mô hình xoáy ốc

- **Ưu điểm**:
  - Linh hoạt, dễ thích ứng với các loại phần mềm và các nhu cầu sử dụng khác nhau, nhất là các phần mềm quy mô lớn
  - Có khá đầy đủ các bước trong tiến trình phát triển, nhất là việc chú trọng phân tích tính rủi ro (risk) của phần mềm cả về mặt kỹ thuật và quản lý
- **Hạn chế**:
  - Phức tạp, cần khá nhiều thời gian để hiểu và vận dụng được một cách hiệu quả
  - Khó khăn trong việc quản lý nhiều chu trình phát triển

Mô hình hướng đối tượng

- Mô hình dựa trên tiếp cận hướng đối tượng, nhằm xác định các đối tượng/lớp của phần mềm;
- Các bước phát triển phần mềm đều xoay quanh việc xác định cấu trúc từng đối tượng, mối quan hệ giữa các đối tượng.

Mô hình hướng đối tượng

- **Ưu điểm**:
  - Hỗ trợ trực tiếp cho việc cài đặt bằng các ngôn ngữ lập trình hướng đối tượng như C++, Java, v.v
  - Tăng cường khả năng tái sử dụng cũng như tính an toàn của phần mềm;
  - Hỗ trợ tốt cho khả năng nâng cấp, sửa chữa phần mềm;
- **Hạn chế**:
  - Phương pháp luận thường khá phức tạp;

Tóm tắt

- Các khái niệm cơ bản
- Các loại phần mềm
- Các quy trình phần mềm phổ biến

Thank you!