ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Nhóm chuyên môn Nhập môn Công nghệ phần mềm

NHẬP MÔN CÔNG NGHỆ PHẦN MỀM

Yêu cầu phần mềm – Khái niệm, phân loại

NỘI DUNG

1. Kỹ nghệ yêu cầu phần mềm
2. Thách thức trong xác định yêu cầu
3. Phân loại yêu cầu phần mềm

MỤC TIÊU

Sau bài học này, người học có thể:

1. Hiểu rõ các khái niệm liên quan tới kỹ nghệ yêu cầu phần mềm (YCPM)
2. Nắm được vị trí và vai trò của kỹ nghệ YCPM
3. Hiểu về quy trình và hoạt động chính của phân tích yêu cầu phần mềm

NỘI DUNG TIẾP THEO

1. Kỹ nghệ yêu cầu phần mềm

1.1. Khái niệm

1.2. Tại sao hiểu rõ yêu cầu khi phát triển phần mềm lại rất quan trọng?

1.3. Nguồn gốc yêu cầu phần mềm

2. Thách thức trong xác định yêu cầu

3. Phân loại yêu cầu phần mềm

1. KỸ NGHỆ YÊU CẦU PHẦN MỀM

1.1. Khái niệm

- Các yêu cầu phần mềm diễn tả chức năng của hệ thống đứng từ quan điểm của khách hàng
  - Các yêu cầu phần mềm cho phép xác định các tính năng (functionality), các ràng buộc và mục đích của hệ thống
  - Yêu cầu phần mềm cần được hiểu rõ bởi cả khách hàng và cả những nhà phát triển
- Tài liệu đặc tả yêu cầu phần mềm (Software Requirement Specification – SRS) chính là sản phẩm đầu ra của pha phân tích yêu cầu phần mềm

1. KỸ NGHỆ YÊU CẦU PHẦN MỀM

1.2. Tại sao hiểu rõ yêu cầu khi phát triển phần mềm lại rất quan trọng?

- Nguyên nhân thất bại của các dự án phần mềm

Yêu cầu không đầy đủ	13.1%
Thiếu sự tham gia của người dùng	12.4%
Thiếu tài nguyên	10.6%
Mong đợi phi thực tế của khách hàng	9.9%
Thiếu sự hỗ trợ nghiệp vụ bên phía khách hàng	9.3%
Thay đổi yêu cầu và đặc tả	8.8%
Thiếu kế hoạch	8.1%
Hệ thống bị lỗi thời	7.5%

*Việc **hiểu sai** yêu cầu của khách hàng sẽ dẫn tới việc xây dựng một hệ thống SAI!*

1. KỸ NGHỆ YÊU CẦU PHẦN MỀM

1.2. Tại sao hiểu rõ yêu cầu khi phát triển phần mềm lại rất quan trọng?

- Requirement Dilemma – Vấn đề nan giải về yêu cầu phần mềm

"Chúng ta không thể xây dựng được một hệ thống trừ khi chúng ta biết chính xác những gì được yêu cầu

Tuy nhiên…

Khách hàng có thể chỉ hiểu biết một phần về hệ thống họ muốn xây dựng"

1. KỸ NGHỆ YÊU CẦU PHẦN MỀM

1.3. Nguồn gốc yêu cầu phần mềm

- Khách hàng – Người sử dụng (Client – User)
  - Khách hàng cung cấp các yêu cầu nghiệp vụ: cung cấp thông tin về công ty, tổ chức trong đó hệ thống sẽ được vận hành, đặc điểm và các quy định, các điều luật riêng của công ty (privacy)
  - Khách hàng cung cấp yêu cầu người dùng (user requirement): cung cấp thông tin về các chức năng cụ thể mà hệ thống sẽ cung cấp
- Đội phát triển cần làm việc với cả hai nhóm khách hàng nói trên để có thể thu thập được đầy đủ thông tin về yêu cầu của hệ thống

NỘI DUNG TIẾP THEO

1. Kỹ nghệ yêu cầu phần mềm

2. Thách thức trong xác định yêu cầu

2.1. Đặc điểm của khách hàng

2.2. Phân tích yêu cầu trong mô hình cồng kềnh (heavyweight)

2.3. Phân tích yêu cầu trong mô hình nhẹ (lightweight)

2.4. Thảo luận

3. Phân loại yêu cầu phần mềm

2. THÁCH THỨC TRONG XÁC ĐỊNH YCPM

2.1. Đặc điểm của khách hàng

- Khách hàng không phải là những nhà phát triển!
- Khách hàng chỉ có những ý tưởng mơ hồ về phần mềm cần xây dựng
- Khách hàng thường hay thay đổi về yêu cầu đối với hệ thống
  - Khi họ nhìn thấy hệ thống, họ sẽ yêu cầu những tính năng mới
  - Những thay đổi đó có thể dẫn tới việc phải sửa chữa hệ thống rất nhiều
  - Những thay đổi này gây ra các vấn đề cho các mô hình phát triển phần mềm

2. THÁCH THỨC TRONG XÁC ĐỊNH YCPM

2.2. Phân tích yêu cầu trong mô hình cồng kềnh (heavyweight)

- Các mô hình Heavyweight như *mô hình thác nước* yêu cầu những đặc tả rất chi tiết về yêu cầu phần mềm
  - Tài liệu đặc tả từng yêu cầu ở mức độ rất chi tiết
  - Tài liệu được sử dụng như một bản hợp đồng giữa hai bên
  - Tài liệu đặc tả được sử dụng trong giai đoạn kiểm thử chấp nhận sau này (acceptance testing)

2. THÁCH THỨC TRONG XÁC ĐỊNH YCPM

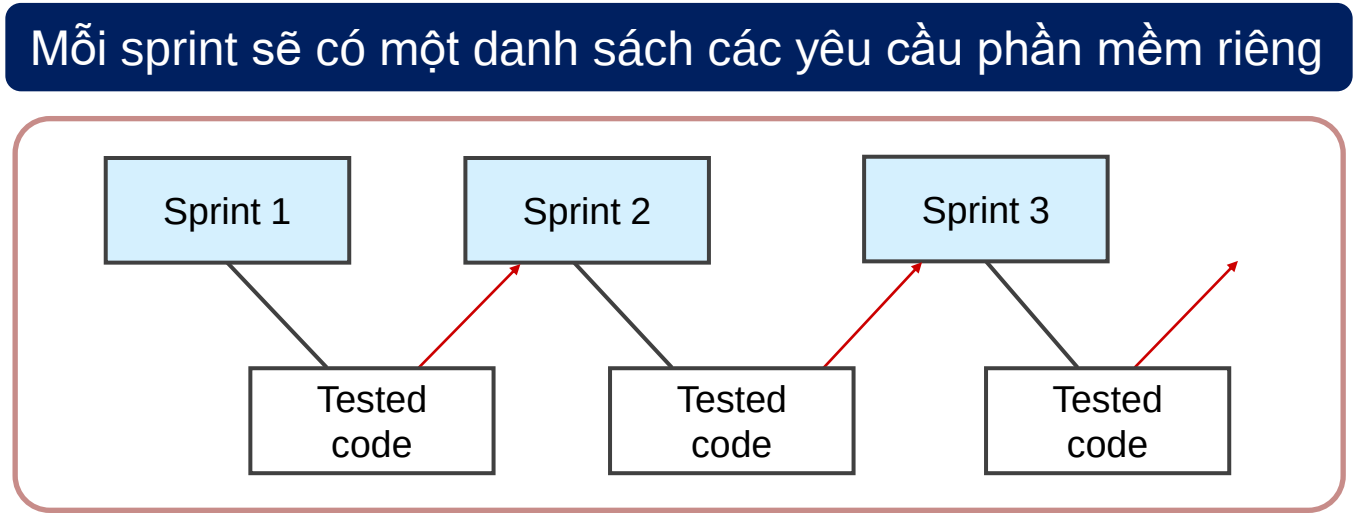
2.2. Phân tích yêu cầu trong mô hình cồng kềnh (heavyweight)

- Khó khăn đối với các mô hình Heavyweight
  - Xây dựng tài liệu đặc tả tốn rất nhiều thời gian và công sức
  - Khó bảo trì hay theo vết thay đổi đối với các đặc tả
  - Kiểm chứng chi tiết từng đặc tả cũng là một công việc tốn nhiều thời gian và công sức
  - Khách hàng thường không hiểu về quá trình xây dựng đặc tả và những tác động của nó lên toàn bộ quá trình phát triển nên họ có thể có những đòi hỏi vô lý

2. THÁCH THỨC TRONG XÁC ĐỊNH YCPM

2.3. Phân tích yêu cầu trong mô hình nhẹ (lightweight)

- Những khó khăn trong việc định nghĩa, xây dựng và bảo trì đặc tả yêu cầu ở mức độ chi tiết trong mô hình Heavyweight là một trong những lí do cho rất nhiều công ty phát triển lựa chọn mô hình Lightweight (mô hình có tính lặp, mô hình phát triển nhanh)



Hình 2.1 – Pha phát triển xây dựng thành Sprint

2. THÁCH THỨC TRONG XÁC ĐỊNH YCPM

2.3. Phân tích yêu cầu trong mô hình nhẹ (lightweight)

- Mô hình Lightweight phát triển một tập hợp yêu cầu lựa chọn trước trong một sprint
  - Working Code – Mã nguồn phát triển được sử dụng để kiểm tra các yêu cầu
  - Khách hàng và nhà phát triển cùng làm việc trên một yêu cầu cụ thể
  - Tài liệu đặc tả chi tiết của yêu cầu được giảm thiểu, không yêu cầu mức độ chi tiết như mô hình Heavyweight → giảm thiểu chi phí thời gian
  - Tài liệu đầy đủ sẽ được hoàn thiện dần dần trong quá trình phát triển

2. THÁCH THỨC TRONG XÁC ĐỊNH YCPM

2.3. Phân tích yêu cầu trong mô hình Lightweight

- Khó khăn trong mô hình Lightweight:
  - Một số yêu cầu phần mềm có quy mô hệ thống, không thể được định nghĩa và xây dựng trong 1 sprint
    - Ví dụ: database, kiến trúc bảo mật, thiết kế giao diện người dùng toàn hệ thống
  - Các yêu cầu được phát triển trong các sprint trong tương lai có thể đòi hỏi sự thay đổi ở các yêu cầu đã được phát triển trong các sprint trước đó

2. THÁCH THỨC TRONG XÁC ĐỊNH YCPM

2.4. Thảo luận

- Cả hai loại mô hình phát triển phần mềm heavyweight và lightweight đều có các khó khăn riêng đối với yêu cầu phần mềm
- Đối với các hệ thống lớn, đội phát triển cần phải **linh hoạt** xử lý
  - Đối với mô hình heavyweight, có thể đặc tả yêu cầu ở mức độ chi tiết **trung bình** và chuẩn bị các phiên bản **tinh chỉnh chi tiết** hơn trong quá trình phát triển, điều đó giúp giảm thiểu thời gian bỏ ra ban đầu và dễ dàng sửa đổi hơn
  - Khi sử dụng mô hình lightweight, có thể đặt **ưu tiên** cho các yêu cầu có mức độ hệ thống ở **giai đoạn sớm** của chu trình phát triển, làm cơ sở cho các sprint sau này

NỘI DUNG TIẾP THEO

1. Kỹ nghệ yêu cầu phần mềm

2. Thách thức trong xác định yêu cầu

3. Phân loại yêu cầu phần mềm

- 3.1. Phân loại
- 3.2. Yêu cầu chức năng
- 3.3. Yêu cầu phi chức năng
- 3.4. Các ràng buộc khác

3. PHÂN LOẠI YÊU CẦU PHẦN MỀM

3.1. Phân loại

- Theo thành phần phần mềm
  - Yêu cầu phần mềm
  - Yêu cầu phần cứng
  - Yêu cầu dữ liệu
  - Yêu cầu về con người (người dùng cuối, tổ chức, khách hàng)

3. PHÂN LOẠI YÊU CẦU PHẦN MỀM

3.1. Phân loại

- Theo đặc tả phần mềm
  - Yêu cầu chức năng
  - Yêu cầu phi chức năng
  - Các ràng buộc khác

3. PHÂN LOẠI YÊU CẦU PHẦN MỀM

3.2. Yêu cầu chức năng

- Mô tả các chức năng mà hệ thống sẽ thực hiện
- Phụ thuộc vào kiểu phần mềm và mong đợi của người dùng
  - Tương tác giữa phần mềm và môi trường, độc lập với việc cài đặt
- Có thể bao gồm các nội dung: giao dịch (transactions), dữ liệu (data) và giao diện người dùng (user interfaces)

3. PHÂN LOẠI YÊU CẦU PHẦN MỀM

3.2. Yêu cầu chức năng

- Các công cụ đặc tả yêu cầu chức năng tiêu biểu:
  - Biểu đồ luồng dữ liệu (Data Flow Diagram)
  - Máy trạng thái hữu hạn (Finite State Machine)
  - Mạng Petri (petri nets)
  - Các biểu đồ ca sử dụng, biểu đồ hoạt động hoặc biểu đồ luồng nghiệp vụ của UML
  - Kết hợp sử dụng ngôn ngữ tự nhiên để đặc tả chi tiết

3. PHÂN LOẠI YÊU CẦU PHẦN MỀM

3.3. Yêu cầu phi chức năng

- Các yêu cầu phi chức năng không liên quan trực tiếp đến các chức năng hệ thống thực hiện
- Yêu cầu về sản phẩm: hiệu năng, độ tin cậy, tính khả chuyển
- Yêu cầu về tổ chức: bàn giao, đào tạo sử dụng, các quy chuẩn
- Yêu cầu từ bên ngoài: tương tác với các hệ thống ngoài, legacy

3. PHÂN LOẠI YÊU CẦU PHẦN MỀM

3.3. Yêu cầu phi chức năng

- **Ví dụ:** Library Management System – Hệ thống quản lý thư viện có các yêu cầu phi chức năng sau đây:
  - Phần cứng và hệ điều hành: IBM/UNIX
  - Hệ quản trị CSDL: Oracle
  - Ngôn ngữ lập trình: C/C++

3. PHÂN LOẠI YÊU CẦU PHẦN MỀM

3.4. Các ràng buộc khác

- Được đặt ra bởi khách hàng, tổ chức vận hành và môi trường thực thi phần mềm
  - Các quy định do tổ chức đặt ra: quy chuẩn về tài liệu, các yêu cầu đặc biệt trong bàn giao sản phẩm và vận hành sản phẩm
  - Các tiêu chuẩn của tổ chức hoặc các giao thức các điều luật của quốc gia cũng có thể là các ràng buộc phi chức năng

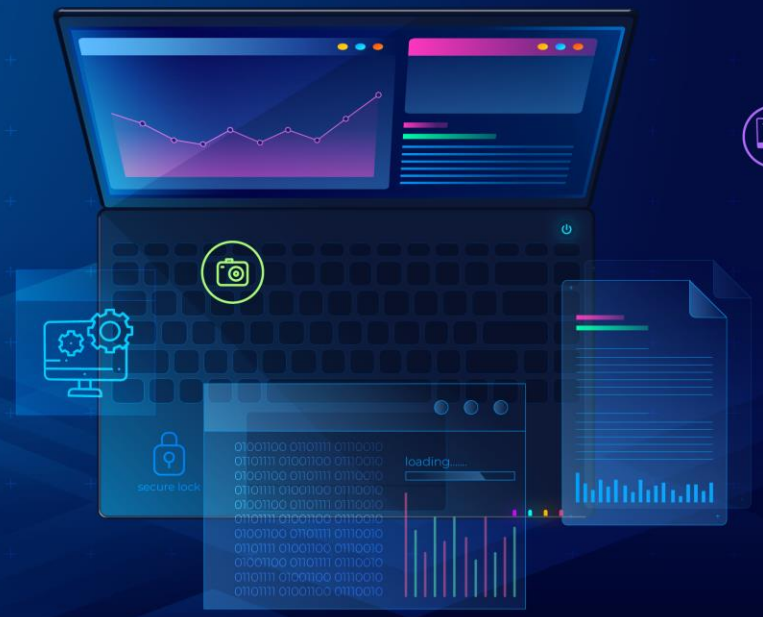
TỔNG KẾT VÀ GỢI MỞ

1. Bài học đã cung cấp cho người học một số **khái niệm cơ bản** trong kỹ nghệ yêu cầu phần mềm (Requirement Engineering)
2. Tiếp sau bài này, **người học có thể tự tìm hiểu thêm** về tài liệu đặc tả yêu cầu phần mềm, các thành phần trong tài liệu này và một số ví dụ có thể tìm được về các tài liệu đặc tả này của các công ty phát triển công nghệ như IBM, Microsoft.

NHẬP MÔN CÔNG NGHỆ PHẦN MỀM

Yêu cầu phần mềm – Khái niệm, phân loại

Biên soạn:
TS. Bùi Thị Mai Anh
Trình bày:
TS. Bùi Thị Mai Anh



NHẬP MÔN CÔNG NGHỆ PHẦN MỀM

Bài học tiếp theo:

Quy trình xây dựng và đặc tả yêu cầu phần mềm

Tài liệu tham khảo

[1] R. Pressman, Software Engineering: A Practitioner's Approach. 8th Ed., McGraw-Hill, 2016.

[2] I. Sommerville, Software Engineering. 10th Ed., AddisonWesley, 2017.

[3] Pankaj Jalote, An Integrated Approach to Software Engineering, 3rd Ed., Springer.

[4] Shari Lawrence Pleeger, Joanne M.Atlee, Software Engineering theory and practice. 4th Ed., Pearson, 2009