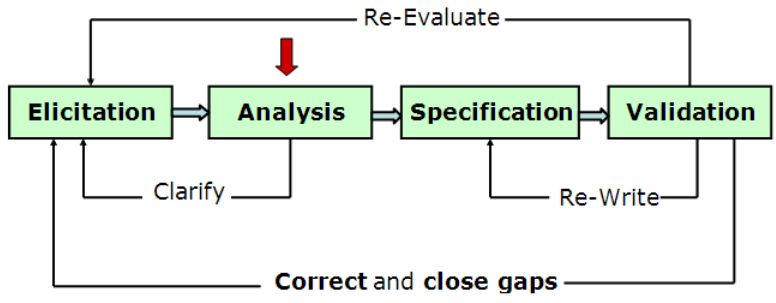
Phân tích và quản lí yêu cầu phần mềm

Phân tích yêu cầu

Lam Quang Vu
Truong Phuoc Loc
References: C1.Ebook +John Vu (CMU)

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Tiến trình phát triển yêu cầu



Iterative process - Multiple-steps, Not sequential

Phân tích yêu cầu

- Là các kĩ thuật quyết định tính năng nào là thích hợp cho sản phẩm dựa trên nhu cầu của stakeholder
- Để phân tích hiệu quả yêu cầu, kĩ sư phần mềm cần hiểu, định nghĩa và xác minh yêu cầu từ quan điểm của các stakeholder để có thể xếp hạng độ ưu tiên các nhu cầu trước khi chỉ định yêu cầu cho phần mềm
- Các yêu cầu phát hiện từ stakeholder phải hoàn thiện và rõ ràng cho việc xác minh sau này

Phân tích yêu cầu

- Tiến trình phân tích nhu cầu của stakeholder nhằmchuẩn bị cho việc định nghĩa hệ thống và yêu cầu phần mềm
- Là tiền trính chuyển các nhu cầu nghiệp vụ thành các mô tả hệ thống với tham số hiệu năng và phân hoạch chúng thành các hệ thống con để cấp phát phần cứng và phần mềm
- Do các stakeholder và nhà phát triển có nhiều quan điểm và cách diễn đạt khác nhau, để hiểu tốt hơn, các kĩ sư phần mềm nên sử dụng các mô tả trừu tượng dễ hiểu và diễn giải

Các quan điểm khác nhau

- Mục tiêu: Định nghĩa hệ thống sẽ làm cái gì

Stakeholders	Developers
Qualitative definition Interpretation to be expected All requests must be met Requirements are evolving On going process Schedule driven System must work	Functional definition Precise, clear Complete Frozen, baseline Must complete within time Task driven "Good" if not perfect system

How do stakeholders and developers communicate?

Phân tích yêu cầu

- Kết quả của phân tích yêu cầu là các mô hình yêu cầu
- Mô hình yêu cầu là các yêu cầu đại diện bởi lược đồ, văn bản có cấu trúc (danh sách, bảng, ma trận)
- Phân tích yêu cầu cũng tập trung vào những thỏa hiệp ra quyết định về tính quan trọng tương đối trong việc xếp hạng độ ưu tiên
- Kĩ sư phần mềm chịu trách nhiệm phân tích các yêu cầu và cộng tác với các stakeholders để xếp hạng độ ưu tiên các nhu cầu
- Là một tiến trình lặp

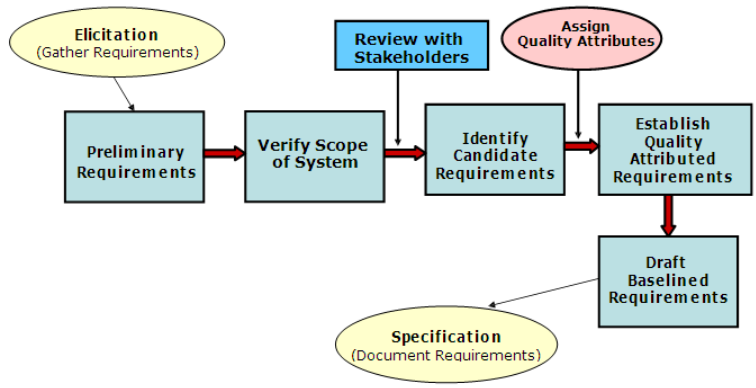
Tại sao phải mô hình hóa yêu cầu?

- Giao tiếp tốt hơn giữa người làm kĩ thuật và kinh doanh
- Cho phép các nhóm dự án nhìn vào các khía cạnh khác nhau của yêu cầu người dùng
- Phát hiện ra những yêu cầu còn thiếu, có lỗi, mập mờ hay mâu thuẫn
- Khám phá phụ thuộc lẫn nhau giữa các yêu cầu
- Cho phép stakeholder thấy tất cả các yêu cầu để hiểu tốt hơn

Tiến trình phân tích yêu cầu

- Xem xét tất cả các yêu cầu để đảm bảo chúng khớp với các mục đích và mục tiêu nghiệp vụ
- Định nghĩa phạm vi dự án
- Tạo ra các yêu cầu chi tiết sử dụng nhiều mô hình giúp stakeholder hiểu rõ các nhu cầu của họ
- Xếp hạng độ ưu tiên các yêu cầu
- Tiếp tục phân tích khi có nhiều chi tiết xuất hiện
- Gán các thuộc tính chất lượng khi các yêu cầu được phát hiện và tinh chế

Tiến trình phân tích yêu cầu



Phạm vi & Biên

- Phạm vi của yêu cầu phải được định nghĩa và tất cả các yêu cầu được chỉ định phải hợp lí trong phạm vi đã định nghĩa
- Chỉ chọn các yêu cầu bạn nên và có thể giải thích
- Bất kì yêu cầu nào được phát biểu mà ra ngoài phạm vi cần phải được xác minh và thảo luận với stakeholder để thêm vào hay loại bỏ ra
- Ghi chú: phạm vi chỉ ra “các biên của toàn bộ hệ thống”
- Các ràng buộc có thể giúp thiết lập phạm vi hệ thống (bien)

Chức năng & Phi chức năng

- Các yêu cầu chức năng mô tả hệ thống phải làm cái gì
  - Là những thứ có thể được mô tả bằng use case hoặc được phân tích bằng cách vẽ các biểu đồ tương tác, biểu đồ trạng thái
  - Có thể liên quan đến module độc lập của chương trình
- Các yêu cầu phi chức năng là các ràng buộc toàn hệ thống như là chi phí phát triển, chi phí điều hành, hiệu năng, độ tin cậy, khả năng bảo trì, dễ chuyển đổi sang nền tảng khác v.v.

Ví dụ về phi chức năng

- Yêu cầu giao diện
  - Giao diện hệ thống mới sẽ tương tác thế nào với môi trường của nó?
  - Giao diện người dùng và tính thân thiện với người dùng
  - Giao diện tương tác với các hệ thống khác
- Yêu cầu hiệu năng
  - Giới hạn về thời gian / không gian
  - Khối lượng công việc xử lí, thời gian hồi đáp, thông lượng và không gian lưu trữ có sẵn. Ví dụ “hệ thống phải hỗ trợ 1000 giao tác mỗi giây”

Ngoài yêu cầu chức năng

- Thuộc tính chất lượng (Các yêu cầu phi chức năng)
  - Khó định nghĩa
  - Thường không được đề cập, nhưng được mong đợi bởi stakeholder, và mỗi stakeholder lại có sở thích khác nhau
  - Quan trọng cho xem xét kĩ thuật (trong suất quá trình ra quyết định thiết kế và kiến trúc)
  - Được xem xét bởi tất cả stakeholder trong suốt tiến trình phát triển yêu cầu – không chỉ người dùng
  - Phải đo lường được và xác nhận được

Chất lượng có nghĩa là gì?

- Mỗi người khác nhau nhìn chất lượng ở góc nhìn khác nhau
- Tester kiểm tra lỗi
  - “Nó có chạy mà không crash không?"
- Quality Assurance xem xét có vấn đề hay không
  - “Nó có đáp ứng các đặc tả?"
- Kiến trúc sư kiểm tra thiết kế
  - “Thiết kế của tôi có đáng tin cậy không?"
- Senior Manager hỏi nhân viên marketing:
  - “Làm sao anh biết chúng ta có sản phẩm có chất lượng?"

Một vài thuộc tính chất lượng - 1

- **Tính sẵn sàng**: Nó có sẵn tại nơi và tại lúc tôi cần không?
- **Hiệu quả**: Nó dùng nhiều hay ít tài nguyên?
- **Linh động:** Việc thêm khả năng mới có dễ dàng hay không?
- **Cài đặt**: Việc cài đặt sản phẩm có dễ dàng hay không
- **Tích hợp**: Nó có bảo đảm chống chọi truy cập không hợp lệ, mất dữ liệu không?
- **Khả năng liên vận**: Nó dễ dàng vận hành với các hệ thống khác như thế nào
- **Khả năng bảo trì**: Việc sửa các lỗi và thay đổi dễ dàng ra sao?

Một vài thuộc tính chất lượng - 2

- **Tính chuyển đổi:** Nó có thể làm việc trên nền tảng khác không?
- **Tính tin cậy:** Mất bao lâu thì hệ thống sẽ không thể chạy được nữa?
- **Tính tái sử dụng:** Chúng ta có thể sử dụng các thành phần vào hệ thống khác dễ dàng ra sao?
- **Robustness:** Nó hồi đáp lại các tình huống không có người can dự tốt ra sao?
- **An toàn:** Nó bảo vệ chống lại hư hại tốt ra sao?
- **Có thể kiểm chứng:** Làm sao tôi xác minh là nó được cài đặt đúng đắn?
- **Tính hữu dụng:** Người ta học dùng nó dễ ra sao?

Các thuộc tính chất lượng

Important to Users	Important to Developers
Availability	Maintainability
Efficiency	Portability
Flexibility	Reusability
Integrity	Testability
Reliability	Performance
Interoperability	Scalability
Security	Modifiability

Các ngữ cảnh chất lượng cụ thể

- Giống như use case được dùng để quyết định cách cư xử của hệ thống (chức năng), một ngữ cảnh dựa trên chất lượng thương được dùng để quyết định các thuộc tính chất lượng của một hệ thống
- Xem xét lại các ngữ cảnh đặc biệt bằng cách đặt câu hỏi với mỗi chức năng về:
  - 1) Hiệu năng
  - 2) Tính mở rộng
  - 3) Bảo mật
  - 4) Tính khả dụng
  - 5) Độ tin cậy
- Ví dụ: Một ngữ cảnh đe dọa trong một trường hợp bảo mật, ngữ cảnh thời gian hồi đáp trong một tình huống hiệu năng, một ngữ cảnh xử lí lỗi trong một trường hợp độ tin cậy

Thuộc tính chất lượng: Hiệu năng

- Hiệu năng là khả năng của hệ thống cấp phát tài nguyên theo cách mà sẽ
  - Thỏa mãn yêu cầu về thời gian
  - Cung cấp nhiều mức độ dịch vụ khi có nhiều yêu cầu cạnh tranh nhau, nhu cầu thay đổi và tính khả dụng của tài nguyên thay đổi
- Hiệu năng thường được chỉ định rõ với các thuật ngữ:
  - **Độ trễ**: Thời gian hồi đáp lại một yêu cầu
  - **Thông lượng**: Số lượng sự kiện hoàn thành trong thời gian quan sát
  - **Sức chứa**: Lượng công việc một hệ thống có thể thực hiện mà vẫn thỏa mãn độ trễ và yêu cầu thông lượng (không suy giảm)

Thuộc tính chất lượng: Bảo mật

- Bảo mật có thể được định nghĩa như là:
  - Không có nguy hiểm, an toàn
  - Sự bảo vệ dữ liệu hệ thống khỏi việc phá hủy hoặc thay đổi không phép
- Bảo mật có ba mức cơ bản:
  - **Tuyệt mật**: Bảo vệ khỏi việc đính kèm không phép.
  - **Tích hợp**: Bảo vệ khỏi thay đổi không phép
  - **Khả dụng**: Bảo vệ khỏi từ chối dịch vụ đối với người dùng được cấp phép

Thuộc tính chất lượng: Khả năng chuyển đổi

- Là mức độ phần mềm chạy trên một nền tảng có thể dễ dàng chuyển đổi sang nền tảng khác.
- Cần xem xét
  - Khả năng chuyển đổi khó định lượng
  - Khó lòng tiên đoán tất cả các nền tảng phần mềm sẽ được yêu cầu chạy
  - Khả năng chuyển đổi bị ảnh hưởng chính yếu bởi các lựa chọn thiết kết như là ngôn ngữ, hệ điều hành và các công cụ có sẵn lẫn đã được tiêu chuẩn hóa
  - Các yêu cầu về khả năng chuyển đổi cần phải được gán độ ưu tiên cho những hệ thống cần phải thực thi trên nhiều nền tảng khác nhau trong tương lai gần

Thuộc tính chất lượng: Khả năng sửa đổi

- Khả năng sửa đổi là chi phí thực hiện thay đổi
- **Cần quan tâm:**
  - Cái gì có thể thay đổi: Chức năng, Phần cứng, Hệ thống, Phần mềm.
  - Ai sẽ tạo ra thay đổi: Nhà phát triển, hệ thống, người dùng
  - Khi nào sẽ có thay đổi: trong quá trình phát triển hay bảo trì
  - Các thay đổi có thể được phân loại dựa vào
    - Xác suất – Khả năng nó sẽ xảy ra
    - Tần suất – Bao lâu thì sẽ thay đổi.
    - Sự phụ thuộc – Cách ảnh hưởng lên những điều khác

Thuộc tính chất lượng: Độ tin cậy

- Khả năng của hệ thống cư xử nhất quán theo cách người dùng chấp nhận được khi vận hành trong môi trường dành cho nó. Độ tin cậy có thể được định nghĩa với thuật ngữ phần trăm (99.999%).
- Ý nghĩa khác nhau cho ứng dụng khác nhau:
  - Mạng điện thoại: toàn bộ mạng lưới có thể không thực thi được trung bình không quá 1h mỗi năm, nhưng các hệ thống chuyển mạch độc lập có thể không hoạt động thường xuyên hơn
  - Hệ thống giám sát bệnh nhân: hệ thống có thể không hoạt động tới 1h một năm nhưng trong trường hợp đó các bác sĩ / y tá phải được cảnh báo về việc này. Tình trạng thất bại thường xuyên hơn của các thành phần khác là không được chấp nhận
- Ví dụ về yêu cầu của độ tin cậy: 'Không có nhiều hơn X bugs mỗi per 10KLOC trong suốt quá trình tích hợp và kiểm thử; không có nhiều hơn Y bugs mỗi 10KLOC vẫn còn trong hệ thống sau khi đã triển khai

Thuộc tính chất lượng: Tính hữu dụng

- Rất nhiều hệ thống chỉ thành công về mặt hệ thống (đáp ứng tất cả các yêu cầu chức năng) nhưng không đáp ứng lợi ích như mong đợi bởi vì người dùng thấy chúng
  - Khó hiểu và khó sử dụng
  - Không nhất quán
  - Cần huấn luyện và tái huấn luyện thêm
  - Dễ có lỗi
  - Nản không muốn tìm hiểu thêm

Đo lường Tính hữu dụng

- 5 lĩnh vực để đo lường tính hữu dụng
  - 1) Thời gian để học sử dụng
  - 2) Tốc độ thực thi
  - 3) Mức độ lỗi
  - 4) Duy trì trong một thời gian dài
  - 5) Thỏa mãn chủ quan
- Các tiêu chuẩn này ở mức độ nào đó có thể được đo lường bằng cách quan sát người dùng sử dụng phần mềm hoặc thiết kế thí nghiệm

Thuộc tính Thỏa hiệp

- Một số thuộc tính nhất định có thể mâu thuẫn nhau, kĩ sư phần mềm phải tạo ra "thuộc tính thỏa hiệp" để cân bằng các đặc tính sản phẩm
- Kĩ sư phần mềm và stakeholder phải quyết định thuộc tính nào quan trọng hơn các thuộc tính khác
- Ví dụ:
  - Bạn không thể mong đợi các hệ thống vừa vận hành trên nhiều nền tảng và cũng có tính hữu dụng
  - Các hệ thống phức tạp(Robust) có thể sẽ không nhanh (hiệu quả) do phải kiểm tra và xác minh dữ liệu nhiều hơn
  - Khó lòng kiểm tra hoàn toàn tính toàn vẹn của hệ thống bảo mật cao

Chuyển đổi sang đặc tả kĩ thuật

Quality Attributes Types	Likely Technical Information
Integrity, Robustness, Usability, Interoperability, Safety	Functional Requirements
Availability, Efficiency, Flexibility, Performance, Reliability	System Architecture
Interoperability, Usability	Design Constraints
Flexibility, Maintainability, Portability Reliability, Reusability Portability	Design Guideline
Portability	Implementation Constraints

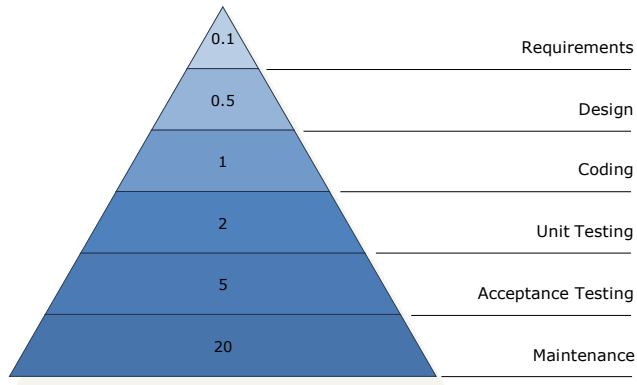
Hoạt động

- Xác định năm thuộc tính chất lượng mà bạn nghĩ quan trọng đối với các phần mềm bạn dùng ngày nay
- Tạo năm câu hỏi về mỗi thuộc tính để làm rõ các kì vọng của mình
- Thử xác định các thuộc tính chất lượng của một trò chơi máy tính mà bạn thích

Vấn đề của các yêu cầu

- Nhiều hệ thống được giao trễ và vượt ngân sách. Chúng thường không thực hiện cái người dùng muốn và thường không tận dụng tối đa được độ hữu ích của mình
- Bước đầu tiên giải quyết bất kì vấn đề nào là xác định nguyên nhân gốc rễ
- Có nghiên cứu đã chỉ ra các nguyên nhân gốc rễ:
  - Thiếu dữ liệu đầu vào từ người dùng
  - Yêu cầu và đặc tả không đầy đủ
  - Thay đổi yêu cầu và đặc tả

Chi phí tích lũy của yêu cầu không tốt / bị bỏ sót



Chi phí tương đối để sửa chữa sai sót

Ảnh hưởng của các vấn đề

- Tái đặc tả
  - Phải định nghĩa lại các yêu cầu
- Thiết kế lại
  - Thay đổi các mô hình và các phụ thuộc
- Lập trình lại
  - Phải viết lại mã nguồn
- Kiểm thử lại
  - Thay đổi kế hoạch kiểm thử, các trường hợp kiểm thử, kiểm thử lại
- Bỏ ra thêm nhiều nỗ lực và mất thời gian / tài nguyên
  - Thu hồi lại / bỏ đi sản phẩm
  - Legal liability / Khách hàng tức giận
  - Tăng chi phí bảo trì
  - Thay đổi các tài liệu

Thế nào là quản lí yêu cầu tốt?

Quản lí yêu cầu tốt

- Ngăn ngừa
  - Chi tiêu quá mức và quá thời gian
  - Hủy dự án
- Các dự án thành công có các "nhân tố thành công" sau
  - Lôi kéo người dùng liên quan
  - Hỗ trợ từ quản lí điều hành
  - Phát biểu các yêu cầu rõ ràng

Chất lượng yêu cầu: Tính nhất quán

- Tính nhấn quán
  - Không có mâu thuẫn
  - Chương trình không hoạt động nếu các yêu cầu mâu thuẫn
- Khó đảm bảo
  - Tập hợp các yêu cầu lớn
  - Các mẫu thuẫn có thể ẩn
    - Ví dụ:
      - Mục 1.2 nói: "không cần nhiều hơn 128MB bộ nhớ"
      - Mục 5.8.9 nói: "hình ảnh nên được dựng theo thời gian thực"
      - => có mâu thuẫn ở đây không???
- Logic hình thức: mọi thứ phải tuân theo các mâu thuẫn
- Vấn đề:
  - Khó long giải thích mâu thuẫn cho khách hàng
  - Khách hàng có thể muốn những thứ bất khả thi

Chất lượng của yêu cầu: Có thể quản lí được

- Các tài nguyên phải đáp ứng các yêu cầu
  - Có thể hoàn thiện đúng giờ không?
  - Với số tiền hiện có?
  - Với các kĩ năng chúng ta có?
- Quản lí rủi ro
  - Các yêu cầu nên được xếp hạng ưu tiên
  - Lí tưởng: có thêm các lựa chọn thay thế
  - Thường không thể nói yêu cầu nào là khả thi
- Quản lí độ phức tạp
  - Đừng làm mọi thứ cùng lúc
  - Tiến trình lặp
  - Tạo ra nguyên mẫu

Chất lượng yêu cầu: Cụ thể

- Chính xác và chi tiết có thể được
- Không tốt:
  - "Chương trình phải nhanh"
  - "Nhận dữ liệu đầu vào là số"
- Tốt:
  - "Chương trình nên hồi đáp nhỏ hơn 1s"
  - "Dữ liệu đầu vào là số nguyên không dấu 16 bit"
- Tạo ra định nghĩa
  - Ví dụ: "Khi nói tới 'Số', có nghĩa là số nguyên có dấu 16-bit"
  - Các thuật ngữ được định nghĩa thường được viết hoa
- Qui tắc định nghĩa
  - Không tham chiếu vòng

Chất lượng yêu cầu: Không có thành kiến cài đặt

- Thành kiến về cài đặt:
  - Cung cấp thông tin về cách thiết kế
  - Cung cấp thông tin về cách lập trình
- Sử dụng thuật ngữ của miền
  - Không dùng thuật ngữ kĩ thuật
  - Bạn muốn tập trung vào WHAT
  - Để HOW sau
- Ví dụ không tốt:
  - "Lưu trữ các sách đã trả vào mảng"
  - "Tính căn bậc hai với thuật toán Newton"
- Ví dụ tốt:
  - "Thư viện biết sách nào đã được trả"
  - "Trả về căn bậc hai với độ chính xác 5 chữ số"

Yêu cầu phải …

- **Đúng**: Phải đại diện nhu cầu thực tiễn của khách hàng và người dùng
- **Đầy đủ**: Bao gồm tất cả các yếu tố cần thiết
  - Chức năng, giao diện bên ngoài, thuộc tính chất lượng và ràng buộc thiết kế
- **Rõ ràng**: Có thể được hiểu giống nhau mà chỉ cần giải thích tối thiểu cho các stakeholder
- **Chính xác**: Được phát biểu đơn giản, theo cách tối thiểu nhất để có thể hiểu được
- **Nhất quán**: không mâu thuẫn với các yêu cầu khác

Yêu cầu cần phải…

- **Hợp lí**: cần thiết để đáp ứng nhu cầu, mục đích, mục tiêu nghiệp vụ
- **Khả thi**: có thể cài đặt được
- **Xác minh được**: có thể dùng kĩ thuật xác định, tiết kiệm chi phí để quyết định yêu cầu đã thỏa hay chưa

Cơ chế phân tích

- Persistency
- Giao tiếp(IPC and RPC)
- Định tuyến thông điệp
- Phân phối
- Quản lí giao tác
- Điều khiển và đồng bộ tiến trình (tranh chấp tài nguyên)
- Trao đổi thông tin, chuyển đổi định dạng
- Bảo mật
- Dò tìm / xử lí / báo cáo lỗi
- Tính dư thừa
- Giao diện truyền thống

Cơ chế phân tích
Các đặc điểm

- Persistency
  - Độ mịn
  - Độ lớn
  - Thời gian
  - Cơ chế đánh giá
  - Độ trễ đánh giá (tạo / xóa, cập nhật, đọc)
  - Độ tin cậy
- Giao tiếp
  - Độ trễ
  - Khả năng đồng bộ
  - Kích thước thông điệp
  - Giao thức

- Giao diện truyền thống
  - Độ trễ
  - Thời gan
  - Cơ chế truy cập
  - Độ trễ truy cập
- Bảo mật
  - Độ mịn của dữ liệu
  - Độ mịn của người dùng
  - Các luật bảo mật
  - Các loại quyền hạn

Hoạt động

- Xác định 5 cơ chế phân tích mà bạn nghĩ quan trọng đối với dự án
- Tạo 5 câu hỏi về mỗi cơ chế để làm rõ kì vọng của bạn