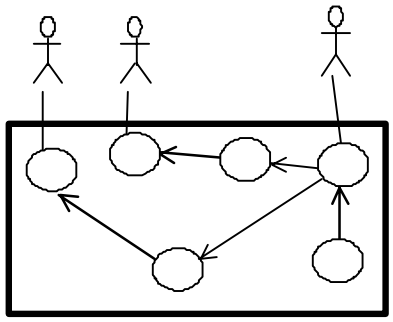
Lecture 10:
Yêu cầu phi chức năng
Non-Functional Requirements (NFRs)

- **Tiền khái niệm:**
  - Các dạng ký pháp mô hình hóa mà chúng ta đã biết
- **Yêu cầu phi chức năng (NFRs) là gì ?**
  - Các hệ số chất lượng, tiêu chuẩn thiết kế; các độ đo
  - Ví dụ về NFRs
- **Tiếp cận hướng sản phẩm (Product-oriented) với NFRs**
  - Tạo ra sự đặc tả các hệ số chất lượng
  - Ví dụ: Sự tin cậy
- **Tiếp cận hướng tiến trình (Process-oriented) với NFRs**
  - Phân tích mục tiêu linh động (softgoal) cho các thỏa hiệp trong thiết kế

Các dạng biểu đồ trong UML…



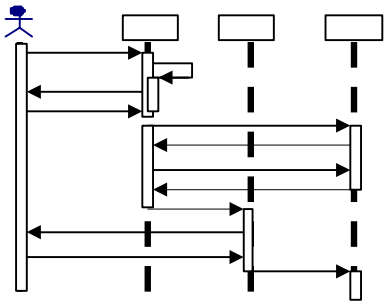
Use Cases

Khía cạnh từ người dùng

Liệt kê trực quan các

chức năng tổng quan của

các yêu cầu chính



Biểu đồ trình tự

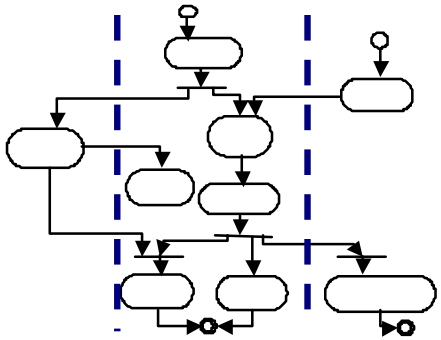
Kịch bản cụ thể

giao tiếp giữa những

người dùng và hệ thống

Trình tự của việc trao

đổi các thông báo



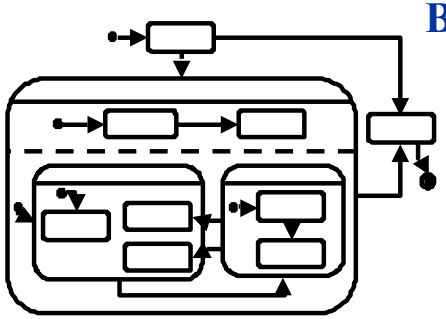
Biểu đồ hoạt động

Tiến trình hoạt động

đồng thời và

đồng bộ phụ thuộc

giữa các công việc



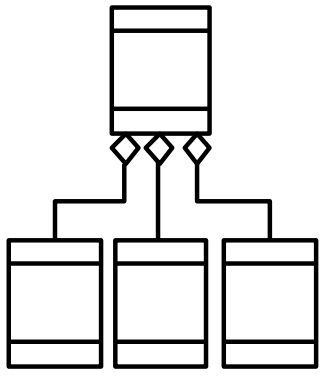
Biểu đồ chuyển trạng thái

Các đáp ứng theo sự

kiện của một đối tượng,

mô hình hóa hành vi

của một lớp giao diện.



Biểu đồ lớp

Cấu trúc thông tin

quan hệ giữa các

lớp, giao diện, hợp tác.

Thể hiện mặt tĩnh

của hệ thống.

…và những biểu đồ không thuộc - UML:

- Mô hình mục tiêu (Goal Models)
  - Nắm bắt các mục tiêu chiến lược của các đối tác
  - Tốt cho việc khảo sát các câu hỏi 'how' và 'why' với các đối tác
  - Tốt để phân tích các thỏa hiệp trade-offs, đặc biệt trên các chọn lựa thiết kế
- Mô hình Cây bắt lỗi (Fault Tree Models) - như một ví dụ trong kỹ thuật phân tích rủi ro
  - Nắm bắt các lỗi tiềm ẩn của một hệ thống và nguồn gốc nguyên nhân
  - Tốt cho phân tích rủi ro, đặc biệt trong những ứng dụng với tiêu chuẩn an toàn
- Mô hình chiến lược phụ thuộc (Strategic Dependency Models (i*))
  - Nắm bắt quan hệ giữa các tác nhân trong một tổ chức
  - Hữu ích cho quan hệ giữa mục tiêu đối tác với tổ chức thiết lập chúng
  - Tốt cho việc thấu hiểu tổ chức sẽ thay đổi như thế nào
- Mô hình quan hệ - thực thể (Enntity-Relationship Models)
  - Nắm bắt quan hệ về cấu trúc thông tin được lưu trữ
  - Tốt cho việc hiểu các ràng buộc và những giả thiết về phạm vi chủ thể
  - Lập nền tảng tốt cho thiết kế CSDL
- Các kiểu bảng lớp (Class Tables), bảng sự kiện (Event Tables) và bảng điều kiện (Condition Tables (SCR))
  - Nắm bắt hành vi động của một hệ thống phản ứng trong thực tế
  - Tốt cho việc biểu diễn chức năng kết hợp từ inputs đến outputs
  - Tốt cho việc tạo các mô hình hành vi chính xác, như suy diễn tự động

Yêu cầu phi chức năng là gì?

Chức năng vs. Phi chức năng

- Các yêu cầu chức năng mô tả cái hệ thống sẽ làm
  - Các chức năng có thể nắm bắt trong use cases
  - Các hành vi có thể được phân tích bằng việc vẽ biểu đồ trình tự, biểu đồ trạng thái, etc.
  - … và khả năng lần vết để giải quyết những vấn đề rắc rối của một chương trình
- Các yêu cầu phi chức năng là những ràng buộc toàn thể trên hệ thống phần mềm
  - e.g. chi phí phát triển, chi phí vận hành, khả năng thực thi, độ tin cậy, khả năng bảo trì, tính khả chuyển, tính thiết thực, etc.
  - Thường được biết như chất lượng phần mềm, hoặc chỉ là "các khả năng" ("ilities")
  - Thường không được cài đặt trong một mô-đun duy nhất của chương trình

Những trở ngại của NFRs

- Khó để mô hình
- Thường ở trạng thái không hình thức, và vì thế mà:
  - Thường mâu thuẫn,
  - Khó thực hiện trong suốt quá trình phát triển
  - Khó đánh giá khách hàng nào ưu tiên để phân phối
- Khó tạo ra các tiêu chuẩn để có thể đo lường chúng
  - Chúng ta muốn ổn định chúng theo cách có thể đo lường được sẽ đáp ứng chúng như thế nào.

Ví dụ về NFRs

Yêu cầu giao diện

- Giao diện của hệ thống mới sẽ như thế nào trong môi trường của nó?
  - Giao diện người dùng "thân thiện"
  - Giao diện đối với các hệ thống khác

Yêu cầu thực thi

- Giới hạn về thời gian / không gian
  - Thời gian tải nạp, thời gian đáp ứng, kích thước dữ liệu nhập và không gian lưu trữ
  - e.g. "hệ thống phải kiểm soát 1000 giao dịch trên giây"
- Độ tin cậy
  - Tính sẵn dùng của các thành phần
  - Sự nguyên vẹn của thông tin dùng duy trì và cung cấp cho hệ thống
  - e.g. "hệ thống phải có ít hơn 1 giờ đình trệ hoạt động trong 3 thángs"
- Tính bảo mật
  - E.g. Cho phép thông tin lưu hành, hoặc phân quyền người dùng
- Khả năng chịu lỗi
  - E.g. Hệ thống sẽ cần năng lực tồn tại, chịu đựng các sự cố tự nhiên, etc

Yêu cầu vận hành

- Các ràng buộc vật lý (kích thước, trọng lượng),
- Mức kỹ năng & khả năng nhân sự
- Dễ bảo trì
- Các điều kiện về môi trường
- etc

Yêu cầu chu trình sống

- "Future-proofing"
  - Khả năng bảo trì
  - Khả năng mở rộng
  - Tính khả chuyển
  - Thị trường mong đợi hoặc vòng đời sản phẩm
- Những giới hạn phát triển
  - E.g giới hạn thời gian phát triển,
  - Tài nguyên sẵn dùng
  - Các chuẩn về phương pháp
  - etc.

Yêu cầu kinh tế

- e.g. giới hạn nghiêm ngặt đúng thời gian và/hoặc vốn dài hạn.

Tiếp cận NFRs

Sản phẩm vs. Tiến trình?

- Tiếp cận hướng sản phẩm (Product-oriented Approaches)
  - Tập trung vào chất lượng của hệ thống (hoặc phần mềm)
  - Nắm bắt các tiểu chuẩn thiết lập của mỗi yêu cầu
  - … để mà chúng ta có thể đo lường chúng khi sản phẩm được thiết kế
- Tiếp cận hướng tiến trình (process-oriented Approaches)
  - Tập trung vào các yêu cầu phi chức năng (NFRs) nào có thể dùng trong tiến trình thiết kế
  - Phân tích tương tác giữa NFRs và các chọn lựa thiết kế
  - … để mà chúng ta có thể đưa ra các quyết định thiết kế phù hợp

Định lượng (Quantitative) vs. Định tính (Qualitative)?

- Tiếp cận định lượng
  - Tìm thang đo các thuộc tính về chất lượng
  - Tính toán mức độ cho một thiết kế đáp ứng với các mục tiêu chất lượng nào
- Tiếp cận định tính
  - Nghiên cứu các dạng quan hệ giữa các mục tiêu chất lượng
  - Lý do của các sự thỏa hiệp (trade-offs), etc.

Chất lượng phần mềm

- Nghĩ đến một đồ vật thông thường
  - e.g. Một cái ghế – bạn sẽ đo "chất lượng" của nó như thế nào?
    - Chất lượng kết cấu? (e.g. độ chắc của những mối nối,…)
    - Giá trị thẩm mỹ? (e.g. tính thanh lịch,…)
    - Đáp ứng mục tiêu? (e.g. sự thoải mái,…)
- Tất cả các độ đo chất lượng đều có quan hệ
  - Không có thang đo nào tuyệt đối
  - Đôi khi chúng ta có thể nói A tốt hơn B…
    - … nhưng thường rất khó để nói tốt hơn thế nào !
- Đối với phần mềm :
  - Chất lượng kết cấu?
    - Phần mềm thì không được chế tạo (mà là phát triển)
  - Giá trị thẩm mỹ?
    - nhưng hầu hết phần mềm thì trực quan
    - giá trị thẩm mỹ là một sự quan tâm bên lề
  - Đáp ứng mục tiêu?
    - Cần phải hiểu rõ mục tiêu

Sự đáp ứng (Fitness)

Source: *Budgen, 1994, pp58-9*

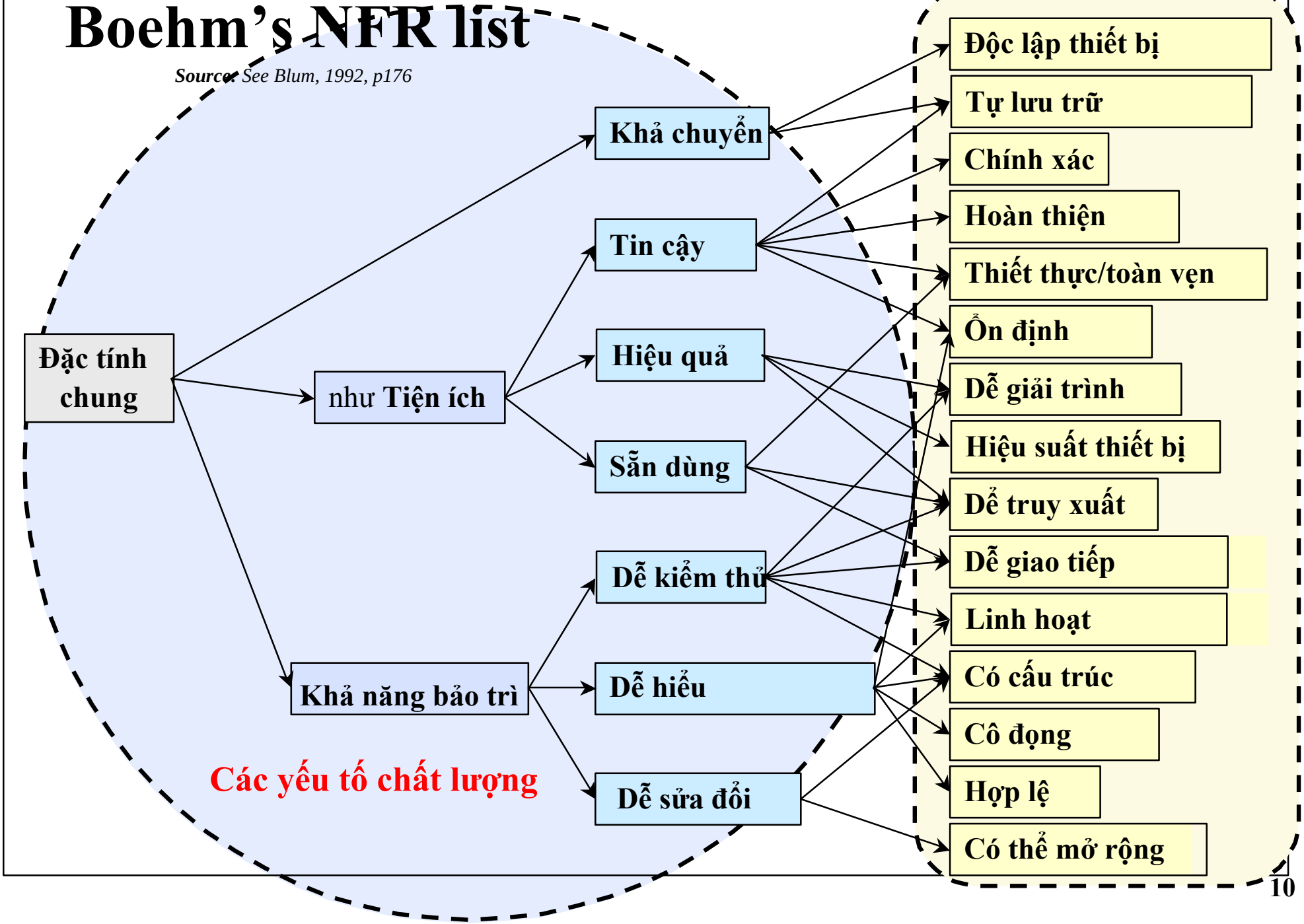
- **Chất lượng phần mềm là đáp ứng với mục tiêu**
  - Nó có thực hiện điều cần thực hiện?
  - Nó có thực hiện theo cách người dùng cần nó thực hiện?
  - Nó có đủ tin cậy? Đủ nhanh? Đủ an toàn? Đủ bảo mật?
  - Nó sẽ có khả năng thực hiện? Nó sẽ luôn sẵn sàng khi người dùng cần đến nó?
  - Nó có thể thay đổi khi nhu cầu thay đổi?
- **Chất lượng không phải là một độ đo cho riêng phần mềm**
  - Nó đo lường các quan hệ của phần mềm trong lĩnh vực ứng dụng của nó
    - không thể đo lường điều này trước khi bạn đặt phần mềm vào môi trường của nó…
    - …và chất lượng sẽ khác nhau trong những môi trường khác nhau!
  - Trong khi thiết kế, chúng ta cần **dự đoán** phần mềm sẽ đáp ứng với mục tiêu tốt như thế nào
    - chúng ta cần những người dự đoán chất lượng tốt (người phân tích thiết kế)
  - Trong khi phân tích yêu cầu, chúng ta cần **hiểu rõ** việc đáp ứng với mục tiêu sẽ được đo lường thế nào
    - Mục tiêu dự định là gì?
    - Các yếu tố chất lượng gì sẽ quan trọng đối với các đối tác?
    - Những yếu tố đó sẽ được tổ chức như thế nào?

Các yêu tố vs. Tiêu chuẩn

- **Các yếu tố chất lượng**
  - Điều này thì liên quan đến quan hệ khách hàng (customer-related)
    - Ví dụ: hiệu năng, tính toàn vẹn, độ tin cậy, tính chính xác, khả năng chịu lỗi, sự tiện dụng,...
- **Tiêu chuẩn thiết kế**
  - Điều này liên quan tới kỹ thuật (hướng phát triển) chẳng hạn như quản lý các bất thường, tính hoàn thiện, tính ổn định, khả năng lưu vết, tính trực quan,...
- **Yếu tố chất lượng và tiêu chuẩn thiết kế thì có liên quan:**
  - Mỗi yếu tố phụ thuộc vào một số tiêu chuẩn liên quan:
    - E.g. Tính chính xác phụ thuộc vào tính hoàn thiện, tính ổn định, khả năng lưu vết,...
    - E.g. Tính khả thi phụ thuộc vào sự mô-đun hóa, tính linh động và tính đơn giản
  - Có thể có một số chuẩn kết hợp để giúp bạn…
- **Trong quá trình phân tích:**
  - Xác định quan hệ quan trọng của mỗi yếu tố chất lượng
    - Từ quan điểm của khách hàng!
  - Xác định tiêu chuẩn thiết kế mà mỗi yếu tố này phụ thuộc
  - Thiết lập độ đo lường các yêu cầu

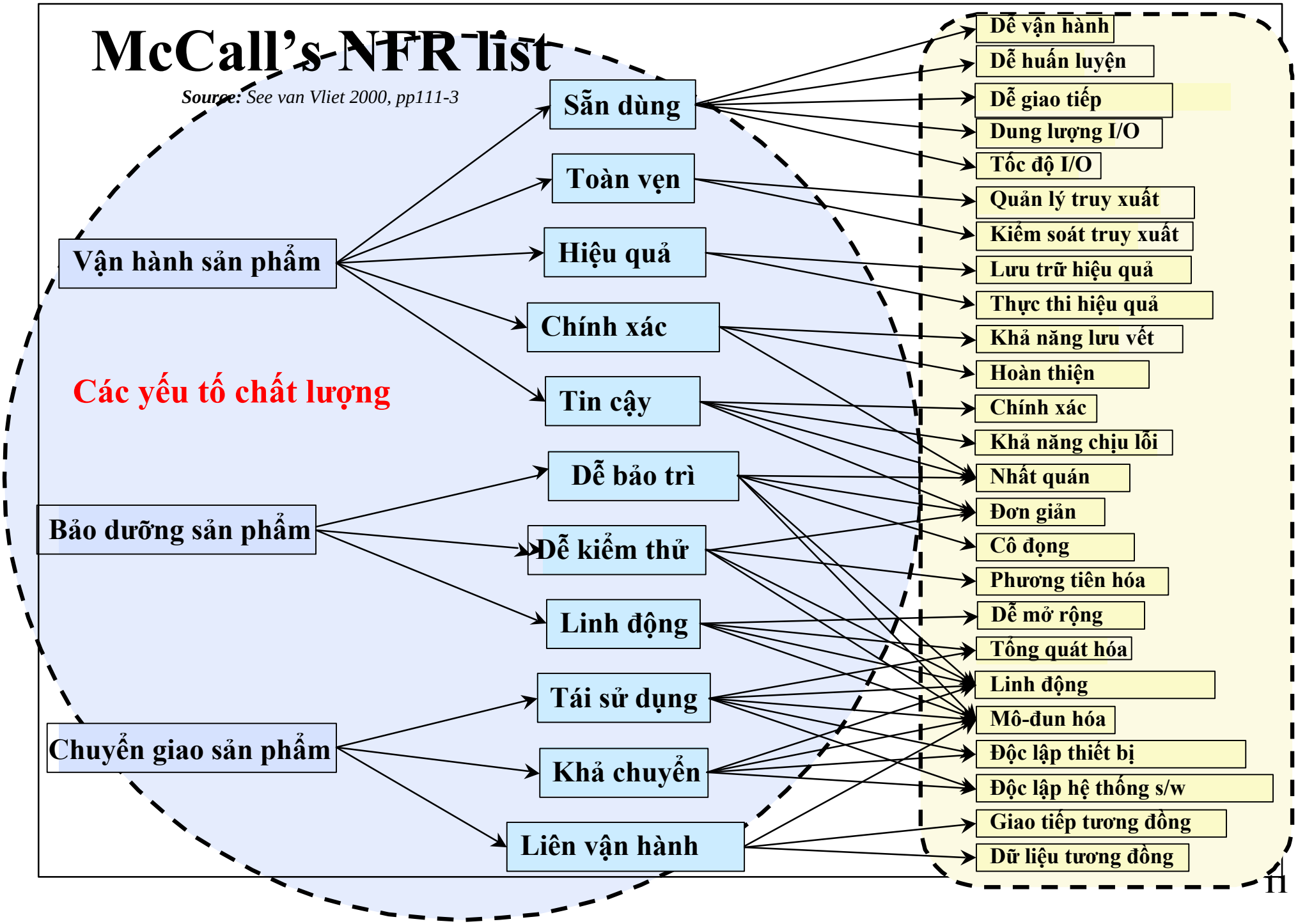
Boehm's NFR list

Source: See Blum, 1992, p176



McCall's NFR list

Source: See van Vliet 2000, pp111-3



Thiết lập độ đo các yêu cầu

Source: *Budgen, 1994, pp60-1*

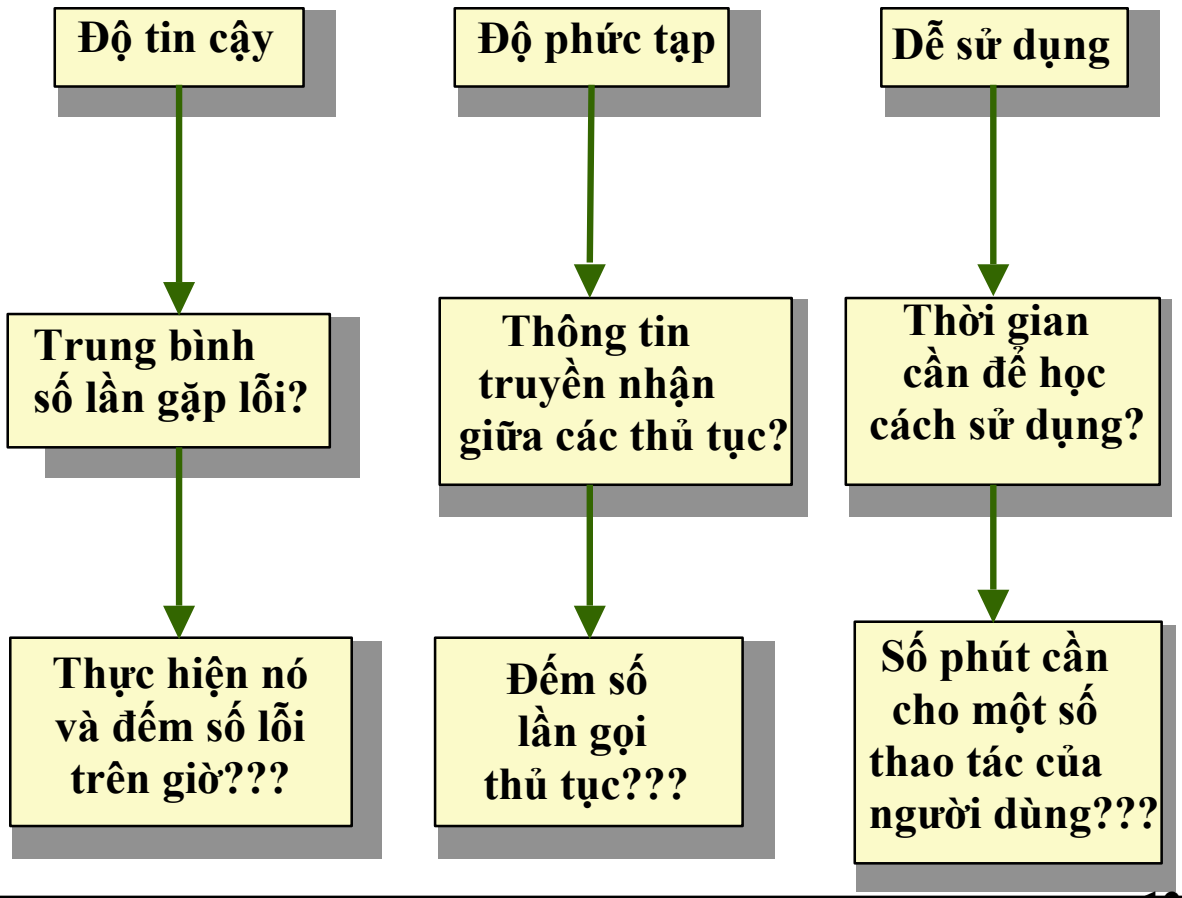
- **Chúng ta phải đổi các khái niệm không rõ ràng về chất lượng thành độ đo lường**

Các yếu tố chất lượng
(các khái niệm trừu tượng
của các đặc tính chất lượng)

Tiêu chuẩn đo lường
(định nghĩa một số độ đo)

Đếm số lần từ kết quả hiển thị của thiết kế
(hiện thực hóa độ đo)

Các ví dụ ...



Ví dụ : Đo độ tin cậy

Định nghĩa ví dụ:

- Khả năng của hệ thống hành xử một cách thống nhất theo phương cách người dùng có thể chấp nhận được khi hệ thống vận hành bên trong môi trường mà nó dự định thực hiện.

Các chú thích:

- Độ tin cậy có thể được xác định dưới dạng một giá trị phần trăm (như, 99.999%)
- Điều này có thể mang ý nghĩa khác nhau đối với các ứng dụng khác nhau:
  - Mạng điện thoại: mạng toàn cục có thể bị lỗi không nhiều hơn, trung bình khoảng 1hr mỗi năm, nhưng lỗi của các chuyển mạng cá nhân có thể xuất hiện thường xuyên hơn nhiều.
  - Hệ thống kiểm tra bệnh nhân: hệ thống có thể bị lỗi đến khoảng 1hr/year, nhưng trong trường hợp này, các bác sĩ/ y tá sẽ báo động lỗi. Lỗi thường xuyên hơn ở các bộ phận cá thể thì không thể chấp nhận được.
- Tốt nhất, chúng ta có thể thực hiện một số việc như sau:
  - "...Không có nhiều hơn X lỗi trên 10KLOC (line of code) có thể được phát hiện trong suốt quá trình tích hợp và kiểm thử; không có nhiều hơn Y lỗi trên 10KLOC có thể tồn tại trong hệ thống sau khi phân phối, theo như tính toán bởi Monte Carlo dùng kỹ thuật tìm kiếm nhân lỗi như trong phụ lục Z; hệ thống phải vận hành 99.9% trên 100% khả năng vận hành theo lịch trong suốt năm vận hành đầu tiên của nó..."

Đo độ tin cậy…

- **Ví dụ - yêu cầu về độ tin cậy:**
  - "Phần mềm sẽ có không nhiều hơn X lỗi trên một ngàn dòng mã lệnh"
  - ... Nhưng có thể đo được lỗi tại thời điểm phân phối sản phẩm không?
- **Dùng trình gỡ lỗi (Debuging)**
  - Đo lường hiệu quả của tiến trình kiểm thử
  - Một số nhân lỗi thì được nêu ra cho hệ thống phần mềm
    - sau đó thực hiện kiểm thử và sửa lỗi không toàn bộ

$$\text{Ước lượng số lỗi trong hệ thống} = \frac{\#\text{ nhân lỗi} \times \#\text{ lỗi phát hiện}}{\#\text{ nhân lỗi được phát hiện}}$$

  - ...NHƯNG, không phải tất cả lỗi đều quan trọng như nhau!

Mô hình ví dụ: Gia tăng độ tin cậy

Source: *Adapted from Pfleeger 1998, p359*

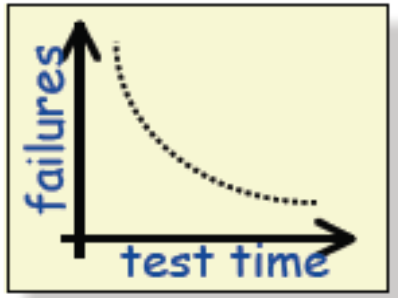
Mô hình kiểm thử Motorola's Zero-failure

- Dự đoán kiểm thử thì cần thiết thế nào khi thiết lập một mục tiêu về độ tin cậy cho trước
- Mô hình cơ sở: hằng số kinh nghiệm (empirical constants)

$$failures = ae^{-b(t)}$$

thời gian kiểm thử (testing time)

Tiến trình đánh giá độ tin cậy



- Inputs cần thiết:
  - fd = mật độ lỗi sau cùng (e.g. 0.03 lỗi trên 1000 LOC)
  - tf = tổng số lỗi kiểm thử quan sát được đến lúc này
  - th = tổng số giờ kiểm thử đến lỗi cuối cùng
- Tính số giờ kiểm thử cần thiết được dùng thêm:

$$\frac{\ln(fd/(0.5 + fd)) \times th}{\ln((0.5 + fd)/(tf + fd))}$$

- Kết quả cho biết số lỗi sẽ phát sinh thêm mà không cần thực hiện số giờ kiểm thử cần thiết để thiết lập mật độ lỗi mong muốn
  - nếu một lỗi xuất hiện vào thời gian này, bạn dừng đồng hồ và tính lại
- Chú ý: mô hình này thì bỏ qua lịch sử vận hành hệ thống!

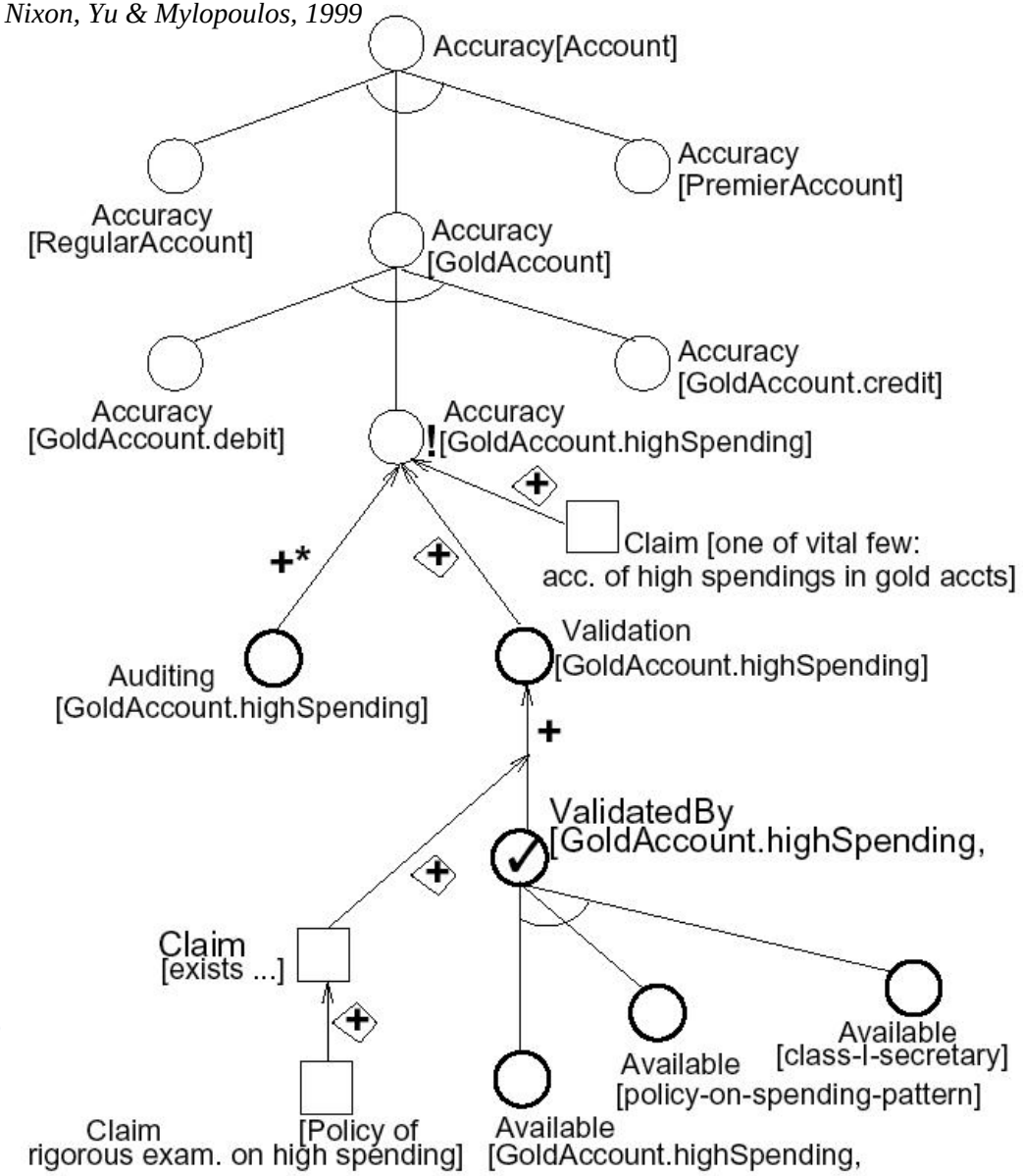
Thiết lập độ đo yêu cầu

- Xác định 'tiêu chuẩn đáp ứng' cho mỗi yêu cầu
  - Đặt 'tiêu chuẩn đáp ứng' bên cạnh yêu cầu
  - E.g. Đối với phần mềm ATM mới
    - Yêu cầu: "Phần mềm phải trực quan và rõ ràng (không cần giải thích gì thêm)"
    - Tiêu chuẩn đáp ứng: "95% các khách hàng hiện có của ngân hàng sẽ có thể rút tiền và gửi séc trong vòng 2 phút khi sử dụng sản phẩm lần đầu tiên"
- Chọn tiêu chuẩn đáp ứng tốt
  - Các đối tác thường hiếm khi mô tả điều này
  - Các tiêu chuẩn đúng không luôn rõ ràng:
    - Những thứ dễ dàng đo được không phải thứ mà các đối tác mong muốn
    - Các độ đo chuẩn thì không phải thứ mà các đối tác mong muốn
  - Làm việc với các đối tác để tìm ra các tiêu chuẩn đáp ứng tốt
- Sự thay thế
  - Đôi khi, chất lượng không thể đo lường trực tiếp. Tìm các định danh thay thế:
    - E.g. "Một vài dữ liệu nhập bị lỗi" thế cho Tính dễ sử dụng
    - E.g. "Liên kết không chặt chẽ" thế cho Tính dễ bảo trì

Sử dụng phân tích softgoal

- **Các dạng goals:**
  - Non-functional Requirement
  - Satisficing Technique
    - e.g. một lựa chọn thiết kế
  - Claim
    - Hỗ trợ/ giải thích sự lựa chọn
- **Các kiểu cấu tạo**
  - Liên kết AND (hợp thành)
  - Liên kết OR (lựa chọn)
  - Liên kết Sup (hỗ trợ)
  - Liên kết Sub (subgoal cần thiết)
- **Đánh giá mục tiêu**
  - Thỏa (Satisficed)
  - Không chấp nhận (Denied)
  - Mâu thuẫn (Conflicting)
  - Không xác định (Undetermined)

Source: *Chung, Nixon, Yu & Mylopoulos, 1999*



Danh mục NFR

Source: *Cysneiros & Yu, 2004*

- **Danh mục định nghĩa sẵn của sự phân tích NFR**
  - **Cung cấp kiến thức nền để kiểm tra độ bao phủ của một NFR**
  - **Cung cấp một công cụ làm rõ NFRs**
  - **Ví dụ:**