

HCI

Quy trình thiết kế

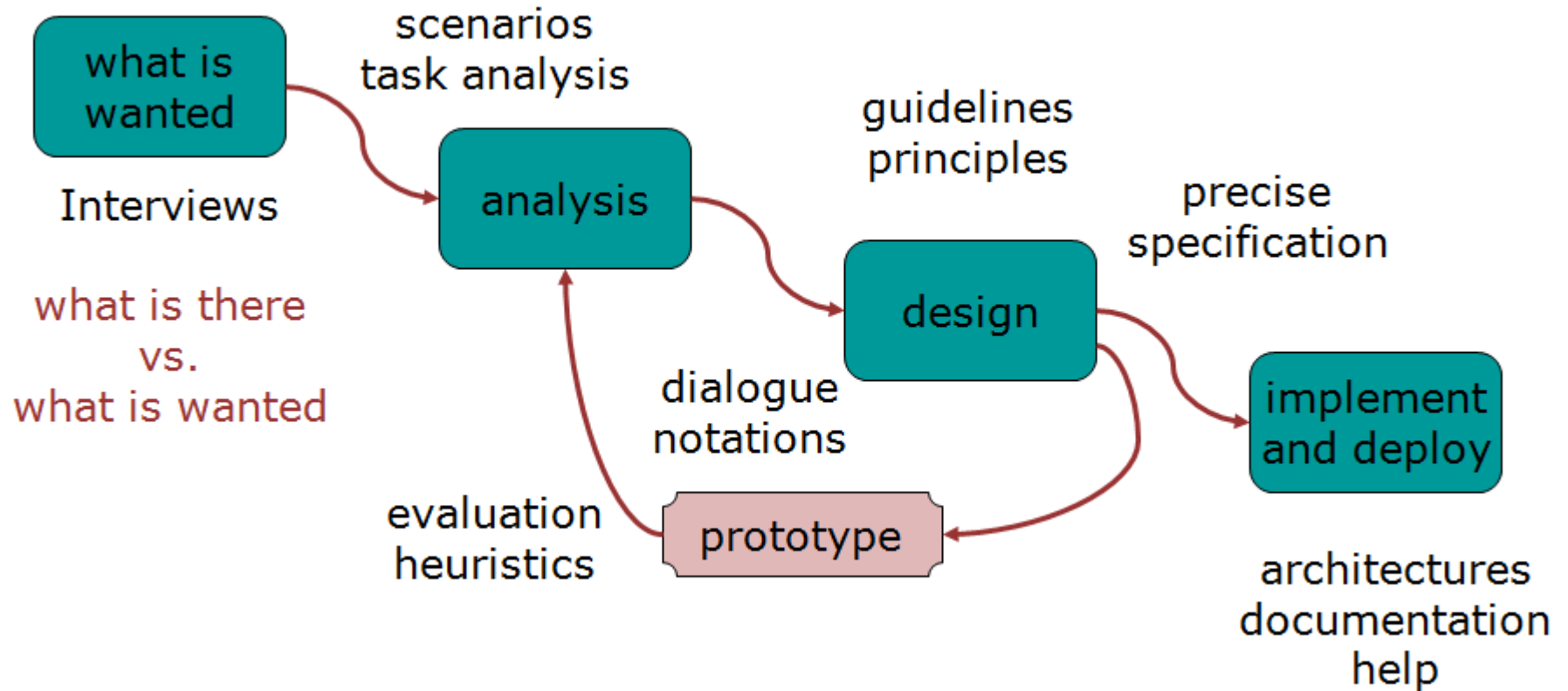
Phạm Thị Ngọc Diễm
Khoa CNTT&TT – ĐHCT

2016-2017

Nội dung

- Quy trình thiết kế giao diện
- Các quy tắc thiết kế

Quy trình thiết kế



[Dix et al, p.195]

Yêu cầu

- Giai đoạn đầu tiên là thiết lập một cách chính xác những cái gì là cái cần thiết
- Cần phân biệt rõ những gì đang có và những gì muốn có
- Một số kỹ thuật sử dụng cho giai đoạn này:
 - Phỏng vấn người dùng,
 - Quay phim họ,
 - Xem xét các tài liệu và các đối tượng mà họ làm việc trên đó,
 - Quan sát trực tiếp.

Phân tích

- Kết quả quan sát và phỏng vấn cần phải được sắp xếp theo một cách nào đó
 - Để đưa ra các vấn đề khóa
 - và giao tiếp với giai đoạn sau của qui trình thiết kế
- Kỹ thuật có thể sử dụng:
 - Kịch bản: những câu chuyện phong phú về sự tương tác, có thể được sử dụng kết hợp với
 - Phân tích công việc
 - Mô hình công việc : một phương tiện để nắm bắt làm thế nào người ta thực hiện các nhiệm vụ/công việc khác nhau
 - = > Những kỹ thuật này có thể được sử dụng để biểu diễn cho cả tình huống hiện có và tình hình mong muốn.

Thiết kế

- Chuyển từ những gì bạn muốn → làm thế nào để làm điều đó.
- Kỹ thuật sử dụng:
 - Có rất nhiều quy tắc, hướng dẫn
 - Các nguyên tắc thiết kế có thể được sử dụng
 - → *Xem lại các chương về các thành phần cơ bản của giao diện, giao diện web.*

Lặp lại và prototype

- Con người rất phức tạp và chúng ta không thể hy vọng có ngay được thiết kế “đúng” ngay lần đầu tiên.
- Vì vậy cần phải đánh giá một thiết kế để
 - Xem nó đang làm việc như thế nào và
 - Chỗ nào có thể cải tiến.
 - → *sử dụng một số kỹ thuật đánh giá.*
 - Một số hình thức đánh giá có thể được thực hiện sử dụng thiết kế trên giấy,
 - nhưng rất khó để có được thông tin phản hồi thực mà không cần có những cố gắng
 - → Hầu hết các thiết kế giao diện người dùng liên quan đến một số hình thức của prototype :
 - sinh ra phiên bản đầu của hệ thống để thử với người dùng.

Cài đặt và triển khai

- Cuối cùng, khi người dùng hài lòng với thiết kế
 - Cần phải tạo ra chúng và triển khai chúng. Điều này sẽ liên quan đến
 - Việc viết mã lệnh / lập trình,
 - Viết tài liệu và hướng dẫn sử dụng
 - Tất cả mọi thứ cần cho một hệ thống thực tế

Quy trình thiết kế giao diện của Galitz

(Galitz, 2007) đề xuất 14 bước trong quy trình thiết kế giao diện người dùng

- 1: Nhận biết ai là người sử dụng
- 2: Hiểu rõ các chức năng nghiệp vụ
- 3: Hiểu rõ nguyên lý thiết kế màn hình và giao diện tốt
- 4: Phát triển menu hệ thống và lược đồ dẫn đường
- 5: Lựa chọn loại cửa sổ thích hợp
- 6: Lựa chọn các thiết bị giao tiếp phù hợp (cho input/output)
- 7: Lựa chọn các điều khiển (controls) trên màn hình phù hợp (button, list, radio button, ...)
- 8: Viết text và thông điệp rõ ràng
- 9: Cung cấp phản hồi, hướng dẫn và hỗ trợ hiệu quả

Quy trình thiết kế giao diện của Galitz

- 10: Cung cấp khả năng quốc tế hóa và khả năng sử dụng rộng rãi (văn hóa, màu sắc, hình ảnh, biểu tượng)
- 11: Tạo các thành phần đồ họa, biểu tượng và hình ảnh có ý nghĩa
- 12: Chọn màu sắc phù hợp
- 13: Tổ chức và bố trí cửa sổ và các trang màn hình
- 14: Kiểm thử hệ thống (prototype, tính khả dụng)

7 QUY TẮC THIẾT KẾ CỦA NORMAN (1988)

Norman's 7 Principles

1. Use both knowledge in the world and knowledge in the head.
2. Simplify the structure of tasks.
3. Make things visible: bridge the gulfs of Execution and Evaluation.
4. Get the mappings right.
5. Exploit the power of constraints, both natural and artificial.
6. Design for error.
7. When all else fails, standardize.

7 QUY TẮC THIẾT KẾ CỦA NORMAN (1988)

- 1. Sử dụng cả "Knowledge in the Head" và "Knowledge in the World"
 - "Knowledge in the Head" : là những kiến thức mà người dùng đã biết
 - "Knowledge in the World": nó là những kiến thức mà người dùng chưa biết đến, và sẽ cần phải được hướng dẫn rõ trên giao diện để người dùng học cách sử dụng.
- 2. Đơn giản hóa cấu trúc của nhiệm vụ/công việc.
 - Nhiệm vụ cần phải được đơn giản để tránh việc giải quyết các vấn đề phức tạp và quá tải bộ nhớ.

7 QUY TẮC THIẾT KẾ CỦA NORMAN (1988)

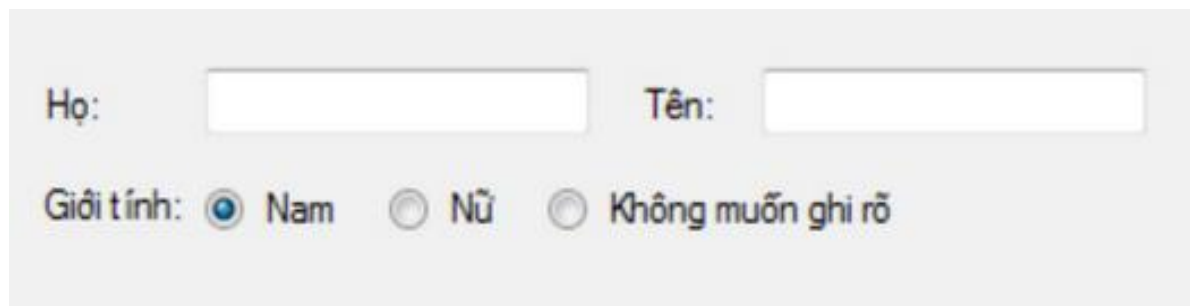
- 3. Làm cho mọi việc có thể nhìn thấy: Thu hẹp khoảng cách giữa thực thi và đánh giá.
 - Giao diện nên làm rõ
 - những gì hệ thống có thể làm và làm thế nào đạt được điều này, và nên
 - cho phép người sử dụng nhìn thấy rõ ràng hiệu quả của hành động của họ trên hệ thống
 - Khoảng cách thực thi (Gulf of Execution): Người dùng cần làm gì để một tính năng có thể chạy được.
 - Khoảng cách đánh giá (Gulf of Evaluation): Làm sao người dùng biết rằng tính năng đó đã chạy theo đúng những gì nó được thiết kế?

7 QUY TẮC THIẾT KẾ CỦA NORMAN (1988)

- 4. Sắp xếp một cách đúng đắn
 - Ý định của người sử dụng nên được phản ánh rõ ràng vào hệ thống điều khiển.
 - Thiết phải theo đúng trình tự và mô hình mà người dùng suy nghĩ đến, khi đó thì việc sử dụng giao diện mới đạt hiệu quả cao.
 - Ví dụ: khi mua đồ trong siêu thị, các bước người dùng sẽ làm đó là xem hàng → xem giá → chọn hàng → tính tiền. Áp dụng điều này vào giao diện của một trang web bán hàng online, thiết kế không thể bắt người dùng xem giá rồi mới xem hàng được.

7 QUY TẮC THIẾT KẾ CỦA NORMAN (1988)

- 5. Khai thác sức mạnh của những ràng buộc, cả tự nhiên và nhân tạo.
 - Ràng buộc là những điều trong thế giới thực mà nó làm cho ta không thể làm bất cứ điều gì *nhưng chỉ làm đúng trong cách đúng*
 - Ví dụ: trong một trò chơi ghép hình, nơi các mảnh chỉ phù hợp với nhau trong một cách.
 - Các ràng buộc vật lý của thiết kế hướng dẫn người sử dụng để làm đúng nhiệm vụ/công việc.



Họ: Tên:

Giới tính: ☒ Nam ☐ Nữ ☐ Không muốn ghi rõ

7 QUY TẮC THIẾT KẾ CỦA NORMAN (1988)

- 6. Thiết kế phải tính đến lỗi
 - Con người là phải mắc lỗi.
 - → Vì vậy phần mềm phải được thiết kế để chịu lỗi từ người dùng, và lỗi do ứng dụng.
 - Khi gặp lỗi → cung cấp cho người dùng một thông báo lỗi hợp lý và có thể bao gồm các thông tin như
 - Cái gì người dùng đã làm
 - Tại sao không đúng
 - Làm thế nào sửa lỗi



7 QUY TẮC THIẾT KẾ CỦA NORMAN (1988)

- 7. Chuẩn hóa khi cần thiết
 - Nếu mỗi phần mềm có một giao diện riêng thì người dùng sẽ rất rối.
 - Vì vậy nếu không có ánh xạ (sắp xếp) tự nhiên thì các ánh xạ tùy ý *cần được chuẩn hóa để người dùng chỉ phải học chúng một lần*
 - Ví dụ: mục đầu tiên trên thanh menu của Hầu hết các phần mềm là menu “File”

8 QUY TẮC VÀNG CỦA Shneiderman

Shneiderman's 8 Golden Rules

1. Strive for consistency
2. Enable frequent users to use shortcuts
3. Offer informative feedback
4. Design dialogs to yield closure
5. Offer error prevention and simple error handling
6. Permit easy reversal of actions
7. Support internal locus of control
8. Reduce short-term memory load

8 QUY TẮC VÀNG CỦA Shneiderman

- 1. Nhất quán
 - Trình tự nhất quán của các hành động nên như nhau trong các tình huống tương tự; các thuật ngữ giống nhau nên được sử dụng trong hướng dẫn, các trình đơn, và màn hình trợ giúp;
- 2. Cho phép người dùng thường xuyên sử dụng các phím tắt
- 3. Cung cấp thông tin phản hồi

8 QUY TẮC VÀNG CỦA Shneiderman

- 4. Design dialogs to yield closure
 - Trình tự của các hành động cần được tổ chức thành các nhóm với một sự khởi đầu, giữa và cuối (begin, middle, and end)
 - Ví dụ với Drag và Drop (Đặng, 2007)
 - Bắt đầu: Người sử dụng nhấn chuột và thấy đối tượng được nhắc lên cùng với con chạy chuột
 - Giữa: Di đối tượng trên màn hình. Phản hồi là đối tượng chuyển dịch
 - Cuối: Nhả phím chuột. Phản hồi ở đây là thấy hiệu ứng nhả đối tượng.
 - Cần có phản hồi thông tin khi hoàn thành một nhóm các hành động và người dùng có thể chuyển đến công việc/nhiệm vụ khác.

8 QUY TẮC VÀNG CỦA Shneiderman

- 5. Cung cấp xử lý lỗi đơn giản
 - Nếu một lỗi xảy ra, hệ thống có thể phát hiện các lỗi và cung cấp cơ chế để xử lý lỗi đơn giản, dễ hiểu.
- 6. Dễ dàng phục hồi các hành động
 - Các đơn vị phục hồi (undo) có thể là
 - một hành động đơn lẻ,
 - một mục dữ liệu,
 - hoặc một nhóm các hành động hoàn thành.

8 QUY TẮC VÀNG CỦA Shneiderman

- 7. Người dùng điều khiển / kiểm soát
 - Thiết kế hệ thống để cho người sử dụng là những người khởi xướng hành động chứ không phải là người đáp ứng.
- 8. Giảm tải bộ nhớ ngắn hạn
 - Giới hạn của việc xử lý thông tin của con người trong bộ nhớ ngắn hạn đòi hỏi
 - màn hình được thiết kế đơn giản,
 - hiển thị nhiều trang thống nhất ,
 - tần số di chuyển cửa sổ giảm,
 - và có đủ thời gian đào tạo được phân bổ cho các code và trình tự của các hành động.