DH2620 - Human-Computer Interaction - Introductory course

Here below are some notes of my reading of the 7th chapter of the course's book Designing for Interaction.

Chap 7 - Refinement

Executing is harder that just creating a concept. Thinking about the details and the aesthetics is the hardest part of the job

The designer has several constraints that end up defining the product: time, money, the business success metrics, user needs, context, tools to maintain the product, team and your own skills.

Laws and Principles of Interaction Design

Direct manipulation

When we can mimic an action that we might perform on a similar object in the physical word (drag and drop, resize window, pushing buttons)

Indirect manipulation

Using in-between command (Delete key) before the development of GUI by Macintosh, most commands were indirect

Feedback

Should occur early and often

The design has to this of how the user should get feedback, a good design provides some mechanism that lets the user know the system has heard the request and is doing something, *this doesn't shorten the waiting time, but it makes it seem shorter*.

Responsiveness of digital products = latency.

Immediate: < 0.1s

Stammer: 0.1s - 1s

Interruption: > 1s ***the attention goes from the task to the product or service itself***

Disruption: > 10s

Feedforward

Knowing what will happen before you perform an action ("Pushing this button will do that"). **Gives *confidence***.

Mental model

User's understanding of how the system works

Standards

Should Ctrl+c always copy whatever is selected?

Designers train users to expect certain items to be located in certain places. A difference to this can cause frustration and annoyance. Alan Cooper's axiom: *Obey standards unless there is a truly superior alternative*.

Fitts's Law

Paul Fitts: The time it takes to move from a starting position to a final target is determined by two things: the distance to the target and the size of the target.

Implications: make buttons reasonably big, use the screen edges since the cursor will slide on the edge (providing infinite width or so...) for menus, and design controls that are close rather that a pull-down menu that involves to travel to other parts of the screen.

Hick's Law

The time it takes for users to make decisions is determined by the number of possible choices they have. People don't consider a group of possible choices one by one. Instead, they subdivide the choices into categories, eliminating about half of the remaining choices with each step in the decision.

Implications: make one menu with all possibilities rather that using hierarchical groups.

The Magical Number 7

We are able to remember 7 plus or minus 2 pieces of information at a time.

Implications: don't design a product that causes cognitive overload, without necessarily reducing the number of elements on your UI to only 7

Tesler's Law of the Conservation of Complexity

There's a point a which you can't reduce complexity even more, but only move the inherent complexity from one place to another.

The Poka-Yoke Principle (avoiding inadvertent errors)

Proper conditions should exist before a process begins. USB flashdrives can only be put in USB ports.

Implications in interaction design: remove functionalities (buttons, icons) if they do nothing in a given state of the system.

Frameworks

Actual or metaphysical structure that defines the product and integrates the content and functionality into a unified whole.

Metaphor

Suggests everything from how a product should function to how its visual or physical form is shaped. The Desktop metaphor has been used by Microsoft to build the elements (Trash, time, folders...). Dashboards and control panels are commonly used metaphors on apps, websites, consumer electronics...

Postures

Different structures for the design of software, depending on how often it should be used and how complex should be the performed tasks.

Structure

Determine the overall form and layout. Using the functional cartography helps. To me it seems like a simple mapping of functionalities (without using programming oriented displays such as UML) that taking into consideration context, priorities, costs, ergonomics, aesthetics and tangibility (tactile). This is doing some information architecture.

Modes

Conditional situations that define which functionalities will be available.

Documentation and Methods of Refinement

Scenarios

The stories such as those we created for our personas. A picture can be worth a thousand words, but a few words can also be worth quite a few pictures, see the very relevant example given in the book. It is a very short scenario, very complicated to storyboard/wireframe/prototype but which took only a few minutes to write. The conclusion given is that: **Using scenarios, designers can sketch with words**.

Sketches and Models

Nothing digital thus far has been able to match the flexibility, speed, and ease of sketching on a piece of paper or whiteboard. Space for example, wall-size whiteboards are better than any computer. Also interesting quote: Sketches have the added bonus of looking unfinished, so no one is inhibited from discussing their flaws.

Storyboards

Technique drawn from filmmaking and advertising, combining a narrative with images and displaying the features of a product/service in a context.

Task flows

Shows tasks in a sensible order. Helps the designer begin to see the product take shape.

Use Cases

Attempts to explain in plain language what a certain function does and why, and involving which actors (personas for example, or even the "system')

Mood boards

Collage that attempts to convey what the final design will feel like, using pictures, colors, words, found in magazines for example. The main purpose is to get inspired. It should emphasize on the emotional part of the final design.

Wireframes

Set of documents that show structure, controls and content. Useful for both designers and other parties, this is actually the most important deliverable after the product itself. Industrial and visual designers, developers, copywriters and business people use wireframes to understand and build the product in a thoughtful way without being distracted by the visual or physical form. It is also for designers to remember where the design decisions came from.

The wireframe should show anything, from an overview of a product to the documentation of one particular functionality. The form of a product contains the content, the controls, and the means of navigating to those.

Content

Includes test, movies, images, icons... if not known yet, use the well-known dummy text *Lorem ipsum*.

Functionality

All the controls, buttons, sliders, check boxes...

Navigation

Hyperlinks, menus, toolbars...

The place of the components on the wireframe should reflect general placement and importance, but doesn't need to be accurate as long as all the items are displayed.

Annotations

Everything that is not labels on the wireframe. An annotation explains the function (of a button for instance), and WHY it is here. The controls and constraints should also be annotated, as well as anything that couldn't be shown in the wireframe itself.

Metadata

The designer name, and several pieces of information that I would call Change Logs. Links to related documents (specs, business requirements...), unsolved issues and a last paragraph where designers can argue to change the constraints.

Service Blueprint

They are key documents for services, composed of service moments and the service sting.

Service moments

Discrete moments that can be designed (example of the sequence of actions to wash a car). There can be multiple designs for each moment.

Service string

Other name for scenario, using storyboards mainly.

Controls

The following describes some basic controls that interaction designers can use.

Switch

Two states: On and Off

Button

Said to be: "Interaction designer's best friend". They're everywhere. It is a switch that is pressed or clicked to activate it. If it stays pressed, we call it a toggle button.

Radio buttons

Used to select an option from a bullet list

Dial

Allows you to use a range of values as an input (rotation dial controlling the heat of an oven)

Latch

Opens an otherwise tightly closed area, like a lock on an armchair.

Slider

Like a Dial but linear

Handle

Protruding part of an object that allows it to be resized or dragged.

Jog dial (only physical world)

Dial that can be manipulated with one single finger.

Joystick (only physical world)

Requires rapid motion, can move in several directions.

Trackball (only physical world)

The base is fixed, but the ball rotates into it and moves in any direction, see both mice models for computers, the standard one (now replaced by optic mouse) and the thumb-roll one, when you control the position of the cursor with your thumb via a rotating ball.

5-way (only physical world)

Button + cursor control. The Macintosh laptops tend to have that today.

Check box (only digital)

Select items from a short list

Twist (only digital)

Little triangles used to display or hide content of a folder in the navigation panels

Scroll bars (only digital)

Move content within a panel

Drop-down menus (only digital)

Clusters navigation options to make it less space-consuming

Multiple-selection list (only digital)

Select multiple items in a list

Text box (only digital)

Enter text or symbols

Spin box (only digital)

Like text boxes but with only a few suggested number values

There are also some non-traditional inputs such as voice-controlled devices, gesture-controlled devices (Kinect, Wii...), the mere presence of a human body that would launch a process

Bill DeRouchey on Frameworks and Controls

From the feedback he gives of his experience, I would keep the following concept: the **hero task**. This is the main task a system has to perform, for example changing volume on a radio, and the design should reflect that hero status. Here the turning volume dial or slider should be accordingly big.