Affective Computing:

The study and development of systems and devices that can recognize, interpret, process, and simulate human affects. Machine should interpret the emotional state of the human, and adapt its behavior to them.

Emotional interaction:

How we feel/react when interacting with technologies

Emotional Design Model:

Reflective: Conscious, intellectually induced reactions.

Behavioral: Expectation induced reactions (subconscious)

Visceral: Perceptually-induced reactions (subconscious)

Anthropomorphism:

Attributing human like characters to inanimate objects. Makes user experience more enjoyable. Some say its deceptive/patronizing.

Interface Types:

1. Command-line: Type phrases at prompts, system responds. Effiecient, precise, fast. Large overhead for learning.
2. WIMP:
   1. Windows (can be scrolled through overlapped ect.) allow more info to be viewed at once.
   2. Icons: Represent objects, applications, commands, tools ect.
   3. Menu: List of options to be scrolled through (dropdown, pop-up, flat-list, scrolling, cascading)
   4. Pointing device: mouse/touchpad controlling cursor

GUI: Same building blocks as WIMPs but more varied. Color, sound, graphical elements like toolbars

1. Virtual Reality- Graphic stimulation presenting synthetic environment
   1. Increased sense of presence
   2. Head mounted display can be uncomfortable
2. Information visualization and dashboard:

Ex. 3-D interactive map

Present data thorugh webs, trees, clusters, scatterplot diagrams, interconnected nodes

Dashboard = how information structures presented to user

1. Speech: talking with system. Siri

6. Touch: fluid and direct. Can be cumbersome, error prone, or slow

7. Robots and Drones: Sociable, remote, domestic, pet robots.

8. Brain computer interface: Provide communication with a device via brainwaves

Prototyping:

Purpose:

1. gain insight into user behavior
2. Communicate ideas to teammates/shareholders
3. Collect data for arguing best design choice

Define questions regarding your design, and answer them

Prototyping process:

1. What are your goals for the prototype

-usability goals and requirments

1. How to measure if goal is achieved. How to measure which prototype is better.

-falsifibility or usability requirements

1. What is the minimum amount of work necessary to produce measure and lear from your prototype.
   1. Strip out features that are not absolutely necessary

Types of prototypes:

Low-fidelity:

Quick, cheap, easy: Story board, sketches, “post-it” notes. Stakeholders can get involved quickly and update the interface.

Limited error checking, poor detailed specification to code to limited usability

Ex. Paper Prototyping, story board, card based, video Prototyping(can be high fidel too)

Wizard of Oz prototyping- User actually talking to designer not computer.(can be low or high fidel)

High-fidelity:

Complete functionality, interactive, have the look and feel of the final project. More resource-intensive to develop, time-consuming, not great for gathering requirments

Uses materials expected in the final product

Compromises:

Horizontal: wide range of functions with little detail

Verticle: Lots of detail for only a few function

Usability laws:

Fitts law:

T=a+blog2(1+(D/W))

Keystroke Level Model:

Assigns average times to common operations. More complex operations are composed of these common operations, sum them up to get the complex operation.

Hicks law:

Describes the time it takes a person to make a decision based on the number of choices available.

T=blog2(n+1)

-diminishing marginal time with respect to input size

Only applies to items that cant be reasonably categorized

Power law of Practice:

Time it takes to do something decreases logarithmically wrt number of trials

Heuristic evaluation:

Small groups of people make list of problems with respect to an interface and assign it a score. They then relate the problem to a heuristic.

Nielsen Heuristics:

1. visibility of system status- always keep user informed, use feedback in a timely fashion.
2. Match between the system and the real world- speak user’s language, make information appear natural.
3. User control and freedom- easy exits, undo and redo
4. Consistent standards- words, situations, and actions should always mean the same thing.
5. Error prevention- limit error prone conditions, warn user of potential errors
6. Recognition rather than recall- Don’t rely on user’s memory
7. Flexibility- System should cater to a wide user base
8. Aesthetic and minimalist design- don’t include irrelevant information
9. Help users recognize, diagnose, and recover from errors- Error messages clear, precisely indicate the problem, offer solution
10. Help and documentation- Provide explicit “help” page