cs121 - software development design

alexandre r.j. français
visiting associate professor of computer science
sources

general design
    the design of everyday things, don norman [Nor02]
    see also: www.jnd.org
    bringing design to software, terry winograd ed. [Win96]

game design
    software engineering for game developers, p. flynt [FS04]
    game design – theory and practice, richard rouse [Rou01]

some material from skip rizzo's introduction to human factors lecture slides (usc)
also:
what is design?
designing for people
   seven stages of action
   knowledge in the head and knowledge in the world
designing for error
   the design challenge
seven principles for transforming difficult tasks into simple ones
game design
what is design?

“whenever objects are created for people to use, design is pervasive”  [Win96]

“more an activity than a thing”

design...

   is conscious, keeps human concerns in the center, is a conversation with materials, is creative (criteria are unconstrained), is communication, has social consequences, is a social activity...

design entails

   intention

   purpose

   target
let's talk about keyboards...

The QWERTY Keyboard

The standard American layout of keys—the widely known QWERTY keyboard.

The Dvorak Keyboard

The American Simplified Keyboard (often called ASK), a simplified version of the original Dvorak keyboard. On the original, the numbers and punctuation keys are arranged differently.

An Alphabetical Keyboard

Most alphabetically organized keyboards arrange the alphabet along horizontal rows, as shown (and in the keyboards of figure 6.5).

Diagonally Alphabetical

This alphabetical arrangement is superior, however, with its diagonal arrangement, letters increase systematically up the alphabet from left to right without major breaks.

A Random Keyboard

Beginners succeed about the same on all these keyboards: alphabetical works barely better than random. For experts, ASK is best, followed by QWERTY; alphabetical keyboards are quite inferior. Moral: Don’t bother with alphabetical keyboards.
"A Man's Busy Day."
Circa 1910
history of the typewriter

1870’s Charles Latham Sholes “Qwerty” keyboard… became the Remington typewriter.

other designs
  circular, piano style (with b&w keys)
  gave way to rectangular board.

no Shift Key, double board

why “qwerty”?
touch typing – Frank McGurrin 1877

many variations in key placement

Dvorak (founder of Industrial Engineering Discipline)
  developed a layout based on painstaking analysis
6.2 Typewriter Keyboards.

The standard American layout of keys—the sheriff at qwerty keyboard.

The American Simplified Keyboard (often called ASK), a simplified version of the original Dvorak keyboard, on the original, the numerals and punctuation keys are arranged differently.

Most alphabetically organized keyboards arrange the alphabet along horizontal rows, as shown (and in the keyboards of figure 6.5).

This alphabetical arrangement is superior, however: with its diagonal arrangement, letters increase systematically up the alphabet from left to right without major breaks.

The keyboard at left has randomly arranged letters.

Beginners succeed about the same on all these keyboards: alphabetical works barely better than random. For experts, ASK is best, followed by qwerty: alphabetical keyboards are quite inferior. Moral: Don’t bother with alphabetical keyboards.
In 1936, August Dvorak, a professor of statistics at University of Washington, proposed rearranging the keyboards alphabetic keys in a layout that is more equitable to the fingers. His design improved efficiency by placing common letters on the home row and also making the stronger fingers do most of the work. The Dvorak layout has been accepted by the American National Standards Institute (ANSI) and has its advocates. However, it has not received wide-spread acceptance because of the many typists trained to use QWERTY keyboards and the costs related to switching over to a new keyboard design.
software design

Software design sits at the crossroads of all the computer disciplines: hardware and software engineering, programming, human factors research, ergonomics. It is the study of the intersection of human, machine, and the various interfaces—physical, sensory, psychological—that connect them.

(definition from the Association for Software Design)
software design facets

software engineering
the discipline concerned with the construction of software
that is efficient, reliable, robust and easy to maintain

interface design
the possibilities of software are both created and
constrained by the physical interfaces

human-computer interaction
whenever someone designs software that interacts with
people, the effects of the design extend beyond the
software itself to include the experiences that people
will have in encountering and using the software
software design facets

art

because computing evolved initially for use in the laboratory and the office, non cognitive aspects have been largely ignored, except by creators of computer games

yet, whenever people experience a piece of software, they have natural human responses: they experience beauty, satisfaction and fun, or the corresponding opposites
designing for people

Many scientists who study artificial intelligence use the mathematics of formal logics – the predicate calculus – as their major tool to simulate thought. But human thought—and its close relatives, problem solving and planning—seem more rooted in past experience than in logical deduction. Mental life is not neat and orderly. It does not proceed smoothly and gracefully in neat, logical form. Instead it hops, skips and jumps its way from idea to idea, tying together things that have no business being put together; forming new creative leaps, new insights and concepts. Human thought is not like logic; it is fundamentally different in kind and in spirit. The difference is neither worse nor better. But it is the difference that leads to creative discovery and to great robustness of behavior.

[Nor02] p.115
principles of design for usability and understandability

conceptual model
make it easy to determine what actions are possible at any moment – make use of constraints

visibility
make things visible, including the conceptual model of the system, the alternative actions, and the result of actions

mapping
follow natural mappings between intentions and the required actions; between actions and the resulting effects; and between the information that is visible and the interpretation of the system state

feedback
make it easy to evaluate the current state of the system, give each action an immediate and obvious effect
seven stages of action

action = goal (1) + execution (3) + evaluation (3)

Forming the goal

Forming the intention

Specifying an action

Executing the action

Perceiving the state of the world

Interpreting the state of the world

Evaluating the outcome
design challenges

gulf of execution
  difference between intentions and allowable actions (affordances)
  example: movie projector vs. VCR – cartridge hides the threading complexity

gulf of evaluation
  amount of effort that the person must exert to interpret the state of the system and to determine how well the expectations and intentions have been met
  example: movie projector vs. VCR – trap door hides state
the seven stages as design aid

Visibility
   by looking, the user can tell the state of the device and the alternatives for action

Good conceptual model
   consistency in the presentation and results
   coherent system image

Good mappings
   possible determine the relationships between actions and results, between the controls and their effects, and between the system state and what is visible

Feedback
   user receives full and continuous feedback about the results of actions.
Fisher-Price Voting Device

If you can’t vote with this you can’t think enough to choose.

Choose Your Shape
4.6 Make the Controls Look and Feel Different. The control-room operators in a nuclear power plant tried to overcome the problem of similar-looking knobs by placing beer-keg handles over them. This is good design, even if after the fact; the operators should be rewarded. (From Seminara, Gonzales, & Parsons, 1977. Photograph courtesy of Joseph L. Seminara.)
knowledge in the head and in the world

precise behavior can emerge from imprecise knowledge for 4 reasons:

information is in the world
great precision is not required
natural constraints are present
cultural constraints are present

behavior is determined by the combination of internal knowledge and external information and constraints trade-off between knowledge in the world and the head
memory as knowledge in the head

memorizing
  arbitrary things
  meaningful relationships
  memory through explanation

short term memory (stm)
  a person should not be required to remember more than 5 or so unrelated items at once

long term memory (ltm)
  information is better and more easily acquired if it makes sense, if it can be integrated into some conceptual framework
  retrieval from ltm is apt to be slow and contain errors

© arf 2010
memory as knowledge in the world

requires physical situation

reminding
  examples: calendar, knotted handkerchief

natural mappings
  example: stove controls layout
designing for error

to err is human!

understand the causes of error and design to minimize those causes
make it possible to reverse actions (undo) or make it harder to do what cannot be reversed
make it easier to discover the errors that do occur, and make them easier to correct
change the attitude towards errors: don't think of the users as making errors, think of the actions as approximations of what is desired
to err is human
to err is human
Dear World,

I just can't take it anymore! I've decided.
http://www.interaction-design.org/misdesigns
the design challenge

Design is an evolutionary, dynamic, iterative process
  non linear
Design must balance aesthetics, usability, cost, schedule, client requirements, etc.
Some reasons why designs might fail:
  Putting aesthetics first (“award winning” designs)
Designers are not typical users
The designer’s clients may not be users

Notice parallel with software...
seven principles for transforming difficult tasks in simple ones

1. use both knowledge in the world and knowledge in the head
2. simplify the structure of tasks
3. make things visible: bridge gulfs of execution and evaluation
4. get the mapping right
5. exploit the power of constraints, both natural and artificial
6. design for errors
7. when all else fails, standardize

[Nor02] p.188
1. use both knowledge in the world and in the head

3 conceptual models:
- design model: what the designer has in mind
- user's model: what the user develops to explain the operations of the system
- system image: physical appearance, operation, way it responds, manual and instructions

ideally, design model \(\equiv\) user's model

user and designer communicate only through system
system image is critical

documentation often not given proper attention
- (1) poor material provided
- (2) people don't read it anyways (human nature...)
2. simplify the structure of tasks

take into account psychology, stm and ltm limitations

provide mental aids
  keep tasks the same

use technology to make visible what would be invisible
  provide feedback and ability to keep control

automate
  but keep task the same
  don't take away control

change the nature of the task
  e.g. hook-and-loop fastener
3. make things visible

bridge the gulfs of execution and evaluation

execution: user knows what is possible and how
evaluation: user can tell the effect of their actions

system should provide actions that match intentions

system should provide indications of system state that
are readily perceivable and interpretable, and that
match intentions and expectations
4. get the mapping right

exploit natural mappings

make sure the user can determine the relationships:
- between intentions and possible actions
- between actions and their effect on the system
- between actual system state and what is perceivable by sight, sound or feel
- between the perceived system state and the needs, intentions and expectations of the user
5. exploit the power of constraints

use constraints so that the user feels there is only one thing to do (the right thing!)
6. design for errors

assume any “error” that can be made will be made
7. when all else fails, standardize

sometimes, an arbitrary choice is necessary
make one and stick with it
game design

“the game design is what determines the gameplay ... in short, the game design determines every detail of how the game is played”  [Rou01] p.xxi

game (creative) design: analogous to a production plan for a film; provides the artistic vision behind the game; implies that software will be developed

game software design: moves forward from the mandate that the production plan offers; reduces the features of the game to a set software requirements and says how the requirements will be implemented

[FS04] p.13
know your customers

players want...
  a challenge, to socialize, a dynamic solitary experience, bragging rights, an emotional experience, to explore, to fantasize, to interact

players expect...
  a consistent world, to understand the game's bounds, reasonable solutions to work, direction, to accomplish tasks incrementally, to be immersed, some setbacks, a fair chance, not to need to repeat themselves, not to get hopelessly stuck, to do – not to watch

players don't know what they want, but they know when its missing

[Rou01] ch.1

© arf 2010
game design document
(sample sections)

Introduction/overview or executive summary
Game mechanics
   what players are allowed to do, how the game is played
Artificial Intelligence
Game elements: characters, items and
   objects/mechanisms
Story overview
Game progression
   events players experience
System menus

[Rou01] ch.19
avoid

class "waffer-thin" or "ellipses" special document
lack of useful content – be specific

the back-story tome
100s pages of story – should be about the gameplay

the overkill document
too much detail is waste – be thorough and consistent
keep to design (avoid requirements or s/w specifications)

the pie-in-the-sky document
keep feet on ground

the fossilized document
keep document up-to-date!