

High level models of human-computer behaviour

Are there theories that describe how people interact with computers?

What is Shneiderman's syntactic/semantic model?

What is Norman's stages of human interaction?

Saul Greenberg

High-level models of human-computer behaviour

Developing Theories in HCI

- must explain and predict human behaviour in the human-computer system
- must work in a wide variety of task situations
- must work within broad spectrum of system designs and implementations

Some low-level theories can be used to predict human performance

- Fitt's law
 - time to select an item with a pointing device
- Keystroke level model
 - sums up times for keystroking, pointing, homing, drawing, thinking and waiting

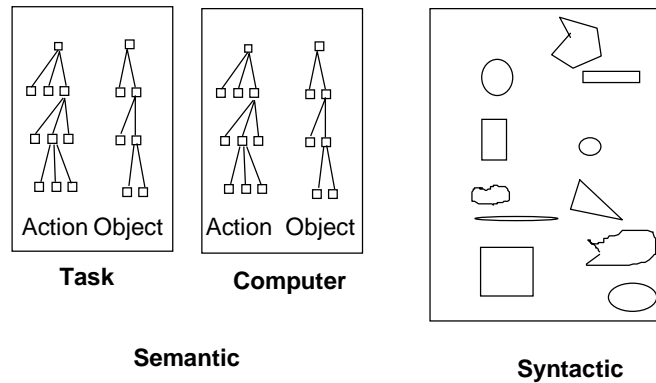
General models that explain human behaviour with machines

- Syntactic/semantic model (Shneiderman)
- Stages of interaction (Norman)
- all of psychology!

Saul Greenberg

Syntactic/semantic model of user knowledge

A high level model of interaction, developed by Ben Shneiderman



Saul Greenberg

1. Syntactic knowledge:

The rules or combinations of commands and signals

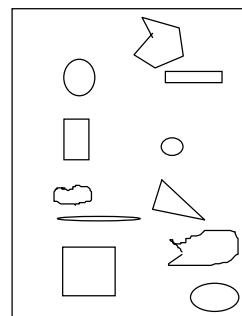
- seen as device-dependent details of how to use system
- examples:
 - backspace key delete previous character
 - right mouse button raise menu
 - grep < word> <file> finding a word in a file
 - tab moves to next field in a form
 - <cntl> X! enlarges window by one line (gmacs)

Saul Greenberg

1. Syntactic knowledge (continued)

User problems with syntactic knowledge

- syntactic details differ between (and within!) systems
 - little consistency→ arbitrary
 - e.g. leaving mail reading in gmacs
 - “q” to quite mail system
 - “<cntl> x <cntl>c” to quit gmacs
 - “<cntl> d or logout” to quit Unix
- hard to learn
 - acquired by rote memorization
 - repeated rehearsals to reach competency
 - must be frequently applied for retention over time
- easily forgotten
 - expert/frequent users ok
 - novice/casual users troubled by syntactic irregularities



Syntactic

Saul Greenberg

2. Semantic knowledge: Computer concepts

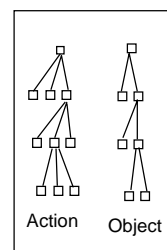
The meaning behind computer concepts

Usually follows a hierarchical structure

- high level concepts decomposed to many low level concepts
- objects
 - e.g. stored information as directories and files as name, length, creation date, owner,...
- actions
 - e.g. saving a file, creating backups, verify access control, etc.

How it works

- people learn computer concepts by
 - meaningful learning
 - demonstrations
 - explanations of features
 - trial by error
 - model of concepts (abstract, concrete, analogical)
 - e.g. file hierarchies are like file/folder systems



Computer

Saul Greenberg

2. Semantic knowledge: Computer concepts

Properties of semantic knowledge (computer concepts)

- relatively stable in memory
 - high level concepts
 - logical structure
 - cognitive model produced
- usually transferable across computer systems
 - but not always!

Problems

- many people now using computers are not computer scientists!
- must be trained in “computer literacy”
- people prefer to concentrate on task, not on computer knowledge

Saul Greenberg

3. Semantic knowledge: task concepts

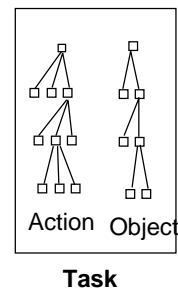
The meaning behind the task concepts

- is independent of the computer

Similar in mechanism to computer concepts

Examples

- how to write a business letter
 - format concerns
 - stylistic concerns
 - paragraph structure, etc.
- creating lecture notes



Saul Greenberg

What Syntactic/Semantic Model reveals

Mapping between three items is extremely important

- Task semantics to computer semantics to computer syntax
 - task semantics: write letter
 - computer semantics: open a file, use editor, save it to disk
 - computer syntax: select menu items, key strokes for formatting,...
- Bad mapping: using latex to write letter
 - aside from task semantics, must also know semantics/syntax of:
 - text editor
 - latex
 - Unix compiling and printing sequence (to typeset and print)
- Relatively good mapping: trashcan to throw away files
 - must know mouse syntax of selecting and dragging
 - computer semantics almost analogous to task semantics

Saul Greenberg

Guideline suggested by syntactic/semantic model

Reduce the burden to the task-oriented user of learning a separate computer semantics and syntax

Methods

- computer semantics
 - metaphors allow computer artifacts to be represented as task artifacts
 - e.g. office workers: files/folders represent hierarchical directory/file systems
 - information hiding
 - don't force people to know computer concepts that are not relevant to their work
- computer syntax
 - A little learning should go a long way...
 - Should be as understandable as possible (tied to semantics)
 - e.g. meaningful command names, icons, keyboard shortcuts
 - Should be as simple as possible and uniformly applicable
 - e.g., object selection with mouse: single click selects, double click activates
 - Generic commands
 - same command can be applied across different objects
 - Syntax should be consistent between systems!

Saul Greenberg

The Four Stages of an Interaction

Intention, Selection, Execution, Evaluation

- a simplified version of Norman's 7 stages

1. Forming an intention

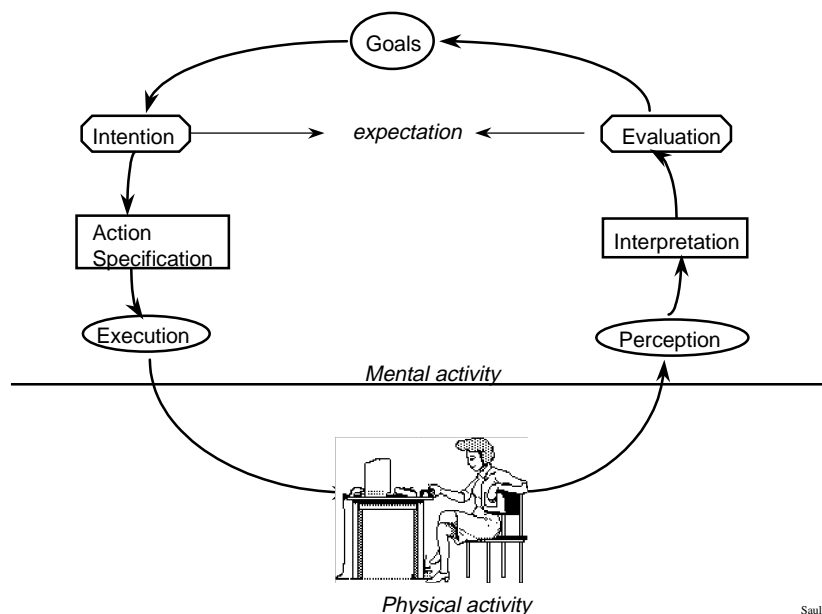
- "What we want to happen"
- internal mental characterization of a goal
- may comprise goals and sub-goals (but rarely are they well planned)
- similar to task semantics
 - e.g. "begin a letter to Aunt Harriet"

2. Selecting an action

- review possible actions and select most appropriate
- similar to mapping between task and compute semantics
 - e.g. "use the emacs editor to create a file harriet.letter"

Saul Greenberg

The stages of user activities when performing a task



Saul Greenberg

The four stages of an interaction continued...

3. Executing the action:

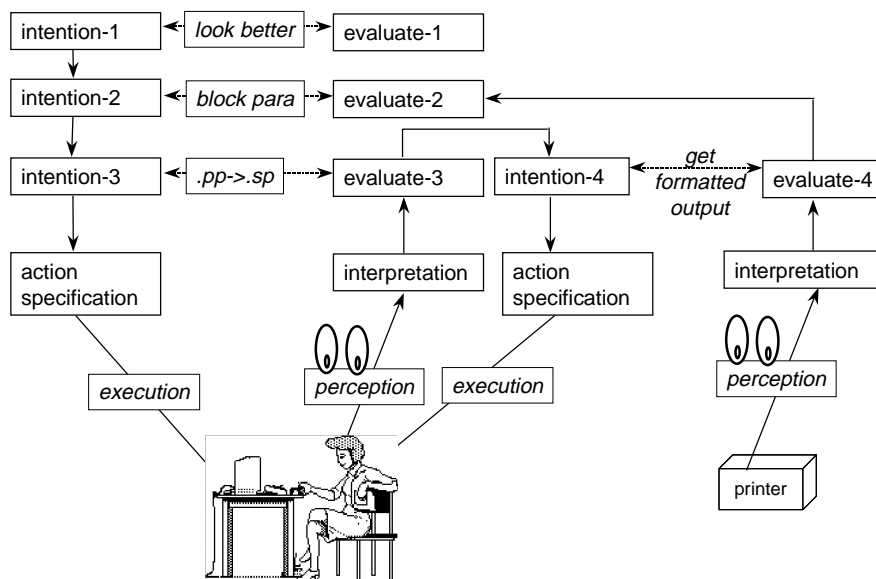
- carry out the action using the computer
- similar to mapping between semantics and computer syntax
 - e.g. type "emacs -nw harriet.letter"

4. Evaluate the outcome

- check the results of executing the action and compare it with the expectations
 - e.g. see if emacs editor is on the display and verify that buffer name is "harriet.letter"
 - requires perception, interpretation, and incremental evaluation

Saul Greenberg

A typical task: making a business letter look better

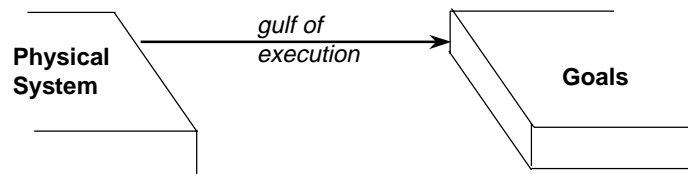


Saul Greenberg

What the four stages model reveals:

The “Gulf of Execution”

- do actions provided by system correspond to the intentions of the user?
- Gulf: amount of effort exerted to transform intentions into selected and executed actions
- A good system:
 - direct mappings between Intention and selections
 - e.g. printing a letter:
 - put document on printer icon
 - vs select print from menu
 - vs “latex letter.tex; lpr -Palw3 latex.dvi”
 - drawing a line: move mouse on graphical display vs “draw (x1, y1, x2, y2)”

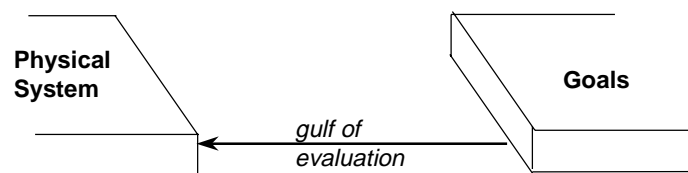


Saul Greenberg

What the four stages model reveals:

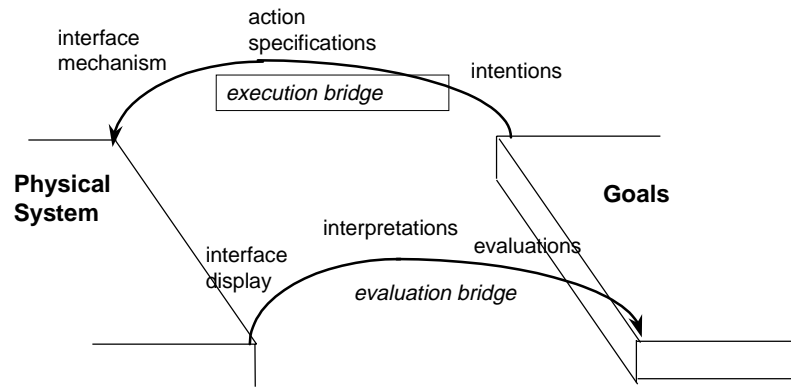
The “Gulf of Evaluation”

- can feedback be interpreted in terms of intentions and expectations?
- Gulf: amount of effort exerted to interpret feedback
- a good system: feedback easily interpreted as task expectations
 - e.g. graphical simulation of text page being printed
- a bad system: no feedback or difficult to interpret feedback
 - e.g. Unix: “\$”, “bus error”, “command not found”



Saul Greenberg

Bridging the Gulf of Execution and Evaluation



Saul Greenberg

Using four stages to ask design questions

How easily can a user

- determine the function of the system?
- tell what actions are possible?
- determine mapping from intention to selection?
- perform the action?
- tell what state the system is in?
- determining mapping from system state to interpretation?
- tell if system is in the desired state?

Saul Greenberg

Using four stages to ask design questions

Questions similar to principles of good design:

- visibility
 - can see state of application and alternatives for actions
- good conceptual model
 - consistency in presentations of operations and results
 - coherent system image
- good mappings
 - relations between
 - actions and results
 - controls and their effects
 - system state and what is visible
- feedback
 - full and continuous feedback about results of actions

Principle of transparency

“the user is able to apply intellect directly to the task;
the tool itself seems to disappear”

Saul Greenberg

You know now

Several high level theories exist that describe how people interact with computers

Shneiderman’s syntactic/semantic model

- a user’s mapping between computer syntax, computer semantics, and task semantics
- problems identified when the user’s mapping is poor

Norman’s stages of human interaction

- intention, selection, execution, evaluation
- problems identified as gulfs of execution and evaluation

Saul Greenberg