High level models of human-computer behaviour

Human Computer Interaction

Based on slide deck
Part 2: Understanding users and their tasks. High level models of human-computer behaviour
Human Computer Interaction I: Principles and Design
by
Saul Greenberg
Professor
University of Calgary, Canada
*The new slides are marked with a **

High level models of human-computer behaviour

Are there theories that describe how people interact with computers?

Quantitative low-level models

- Fitts's Lae
- Keystroke level model
- ...

Qualitative high-level models

- Shneiderman's syntactic/semantic model
- Norman's stages of human interaction
- ...

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

High-level models of human-computer behaviour

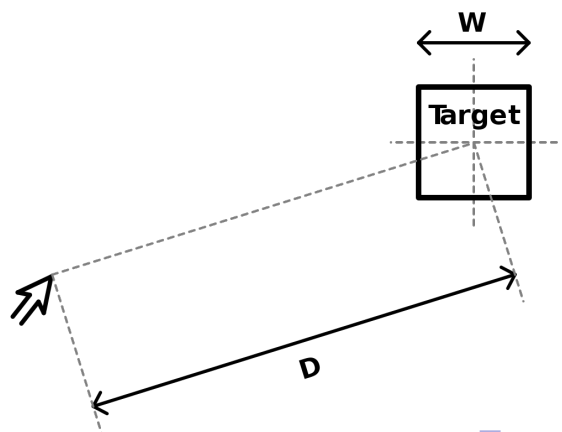
- **Low-level theories can be used to predict human performance**
  - Fitts's law
    - time to select an item with a pointing device
  - Keystroke level model
    - sums up times for keystroking, pointing, homing, drawing, thinking and waiting

*Fitts's Law

- The **time required to rapidly move to a target area is a function of the ratio between the distance to the target and the width of the target**

- used to model the act of *pointing* (physically or virtually)

- $MT = a + b \times log_2(2 \times D/W)$
  - MT - time to hit the target
  - a,b - empirically determined constants



*Keystroke level modelling

- estimates the ideal task time
- based on a detailed analysis of all elementary steps that users need to go through to reach the goal of the task
- it works by summarizing all the elementary steps (keystroke, typing, mouse click, homing, pointing, **mental operations**, system response) which are needed for finalizing the task

*Keystroke level modelling - An example

Assumptions: Hands on keyboard

Design A	Design B
1. **Initiate** the search operation M + 1.2 sec. 2. **Home** mouse H + 0.4 sec. 3. **Point** the mouse to the Searchbox field P + 1.1 sec. 4. **Click** into the Searchbox field BB + 0.2 sec. 5. **Home** keyboard H + 0.4 sec. 6. **Recall** the search term M + 1.2 sec. 7. **Type** /ACCServer.*01|Prod|.*Agent/ 31K + 8.68 sec. 8. **Type** ENTER K + 0.28 sec.	1. **Initiate** the search operation M + 1.2 sec. 2. **Home** mouse H + 0.4 sec. 3. **Point** the mouse to the Searchbox field P + 1.1 sec. 4. **Click** into the Searchbox field BB + 0.2 sec. 5. **Find** the list of Named Filters M + 1.2 sec. 6. **Click** link BB + 0.2 sec. 7. **Find** the Named Filter M + 1.2 sec. 8. **Click** the Named Filter BB + 0.2 sec. 9. **Click** the Search button BB + 0.2 sec.
M + H + P + BB + H + M + 31K + K 1.2 + 0.4 + 1.1 + 0.2 + 0.4 + 1.2 + 8.68 + 0.28	M + H + P + BB + M + BB + M + BB + BB 1.2 + 0.4 + 1.1 + 0.2 + 1.2 + 0.2 + 1.2 + 0.2 + 0.2
13.46 sec.	**8.8 sec.**

Peter Zalman, **Key-stroke Level Model**
https://medium.com/enterprise-ux/key-stroke-level-model-213dd054e5c7

*High-level models of human-computer behaviour

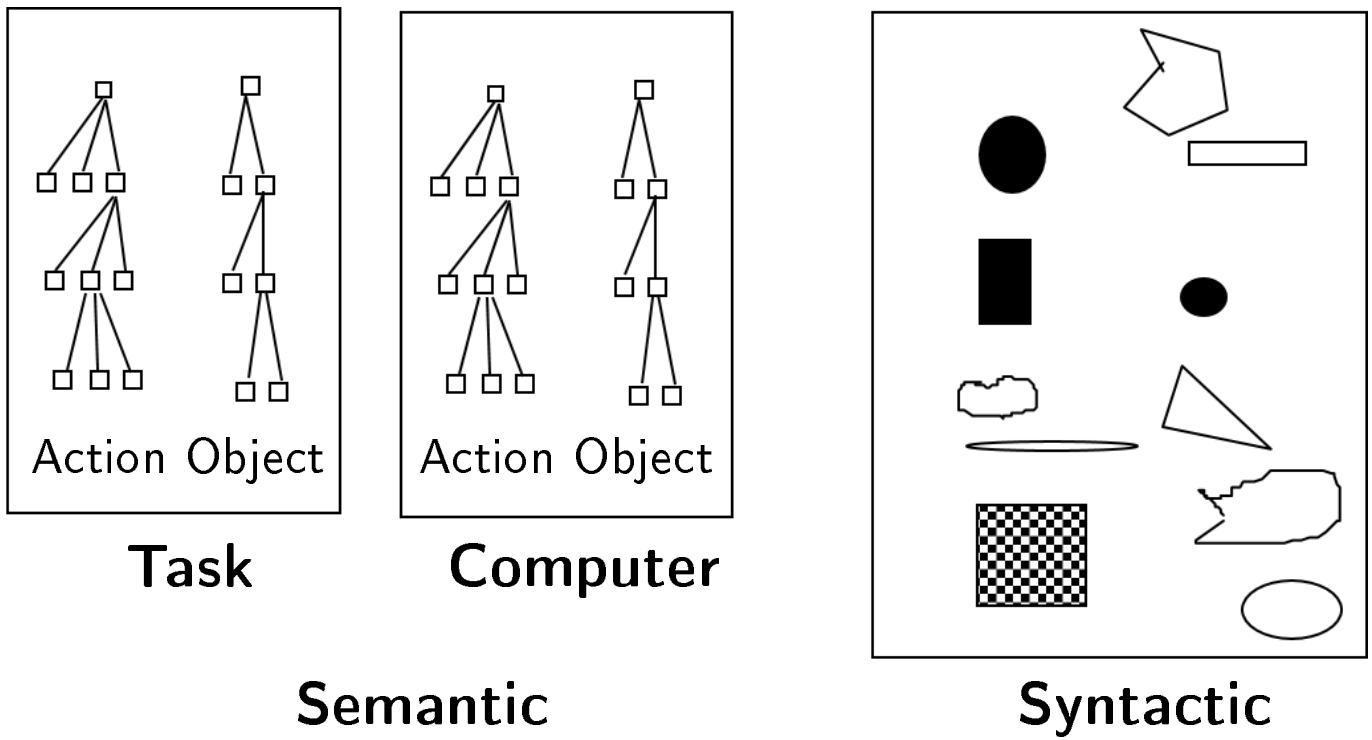
- Uses of these low-level theories
  - compare two design variants
  - common benchmarking
  - measure improvement in case of reduce time-on-task by N% design goals
  - user testing (in case evaluation with real users is not possible)

High-level models of human-computer behaviour

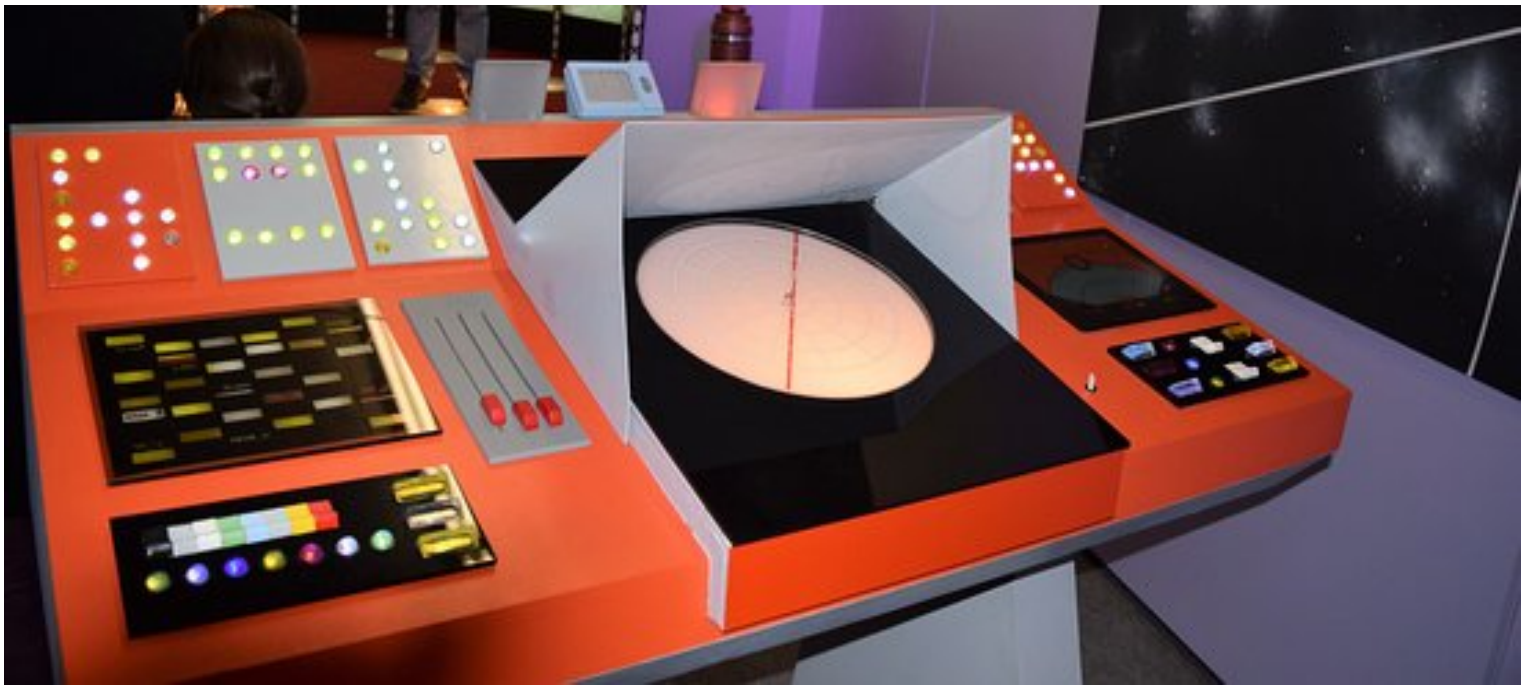
- **General models that explain human behaviour with machines/computers**
  - Syntactic/semantic model (Shneiderman)
  - Stages of interaction (Norman)
  - all of psychology!

Syntactic/semantic model of user knowledge

- A high level model of interaction, developed by Ben Shneiderman



*Syntactic knowledge



Star Trek Transporter console
https://vidasmps.com/star-trek-map/
star-trek-map-lovely-transporter-console-at-the-ready-pictu

Syntactic knowledge

- **The rules or combinations of commands and signals**
  - seen as device-dependent details of how to use system
  - examples:
    - backspace key - deletes previous character
    - right mouse button - raises contextual menu
    - grep <word> <file> - finds a word in a file character
    - tab - moves to next field in a form
    - dd - deletes current line in vim text editor

User problems with syntactic knowledge

- syntactic details differ between (and within!) systems
  - little/arbitrary consistency
  - e.g.
    - :q - quits vim text editor
    - exit - quits current login user
    - Ctrl + X - quits nano text editor
    - q - quits displaying content with less

User problems with syntactic knowledge

- 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

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.

Semantic knowledge: computer concepts

- **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

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

*Semantic knowledge: task concepts

Racing wheel for Playstation
`https://origin-gaming.logitech.com/en-us/product/g29-driving-force`

Semantic knowledge: task concepts

- **The meaning behind task concepts**
  - is independent of the computer

- **Similar in mechanisms to computer concepts**
- e.g.
  - how to write a business letter

    format concerns
    stylistic concerns
    paragraph structure
    etc.
  - creating lecture notes

What syntactic/semantic model reveals

Mapping between three items is extremely important: **Task semantics to computer semantics to computer syntax**

- e.g.
  - task semantics: write letter
  - computer semantics: open a file, use editor, save it to disk
  - computer syntax: select menu items, key strokes for formatting,...

What syntactic/semantic model reveals

- 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

Guidelines suggested by syntactic/semantic model

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

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

Guidelines suggested by syntactic/semantic model

- 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!

The Four Stages of an Interaction

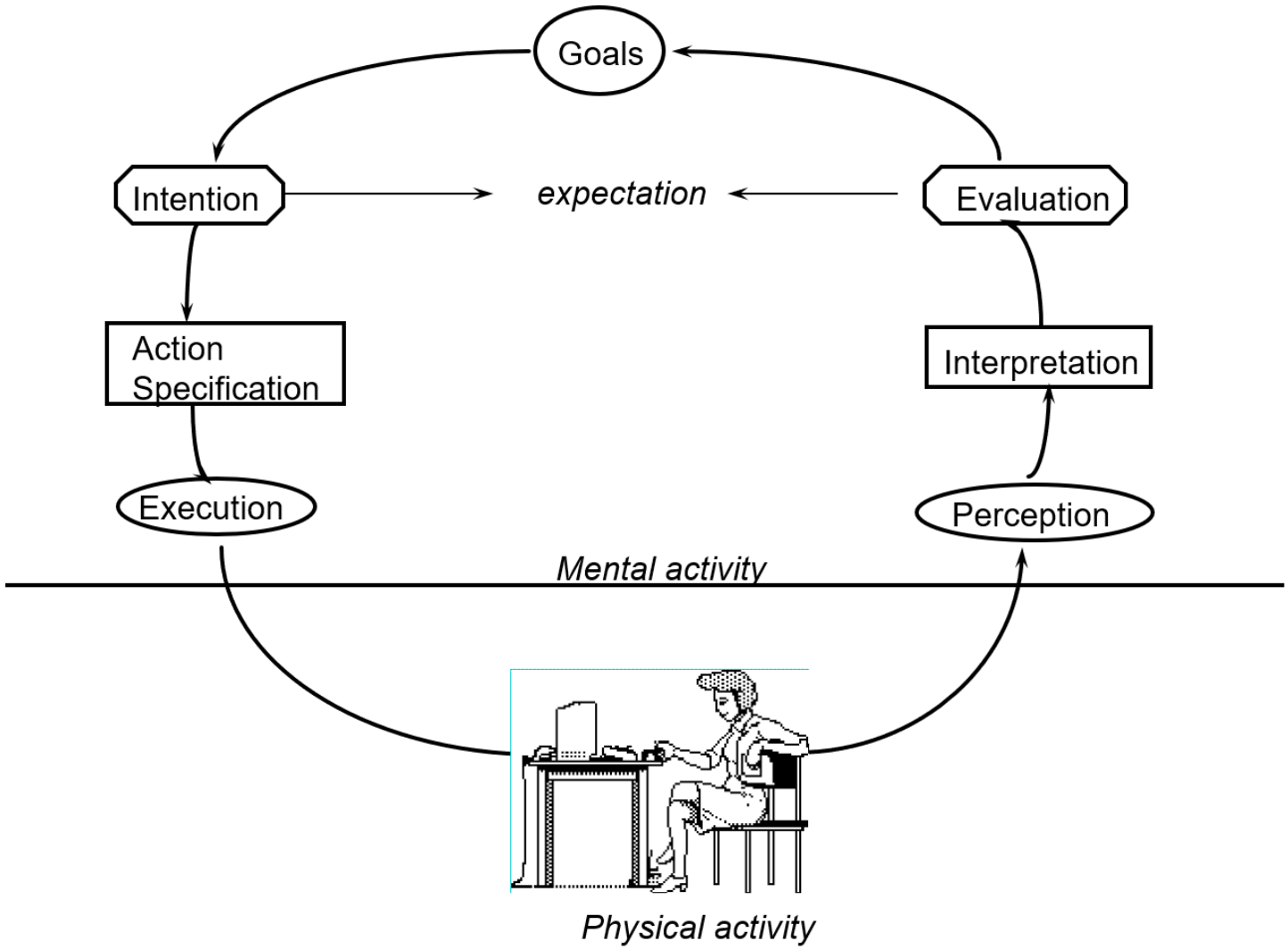
Intention, Selection, Execution, Evaluation (a simplified version of Norman's 7 stages)

- 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 gedit editor to create a file harriet.letter"

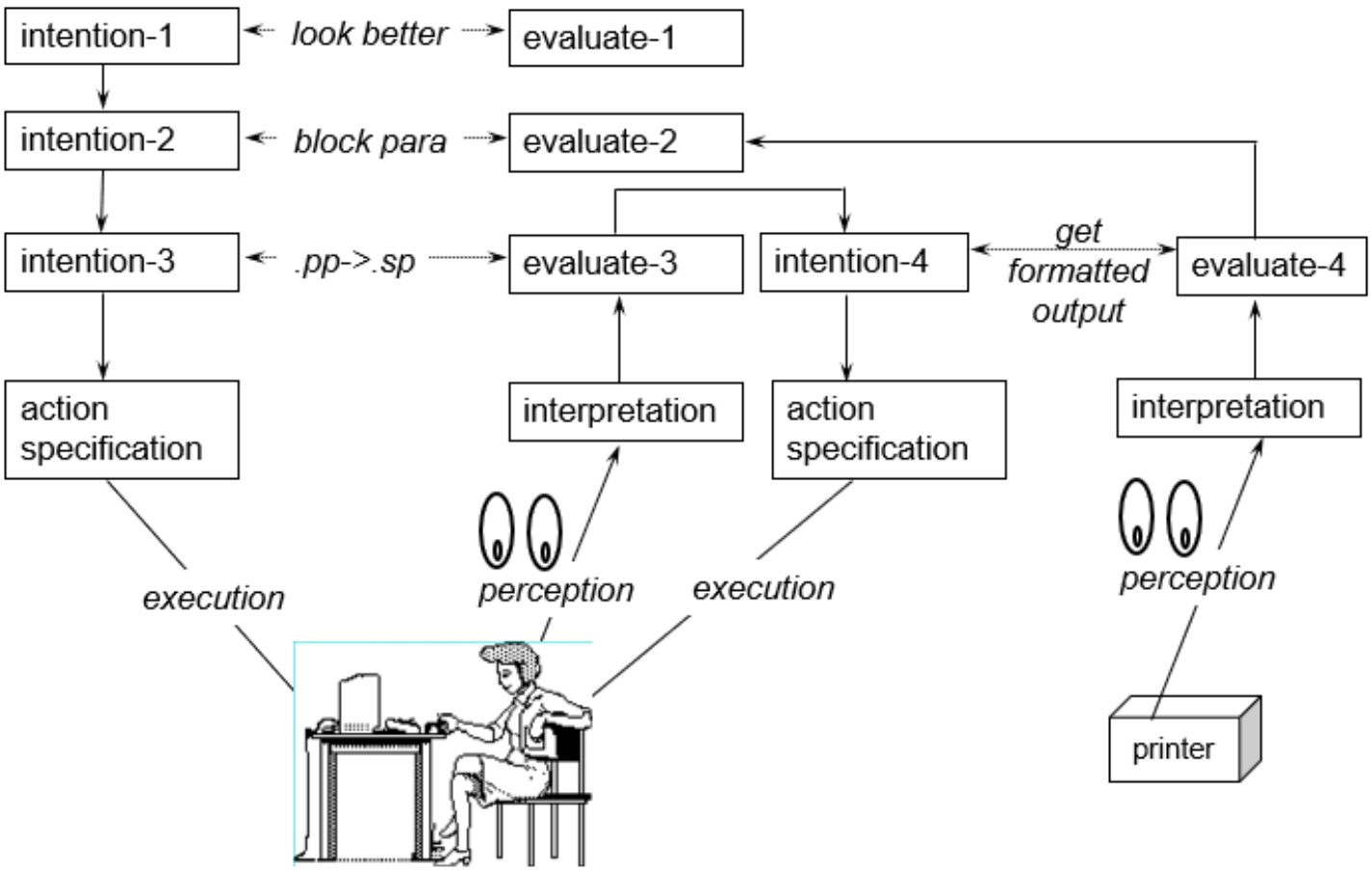
The Four Stages of an Interaction

- 3. **Executing the action**
  - carry out the action using the computer
  - similar to mapping between semantics and computer syntax
    - e.g. type "gedit harriet.letter"
- 4. **Evaluate the outcome**
  - check the results of executing the action and compare it with the expectations
    - e.g. see if gedit editor is on the display and verify that file name is "harriet.letter"
    - requires perception, interpretation, and incremental evaluation

The stages of user activities when performing a task



A typical task: making a business letter look better



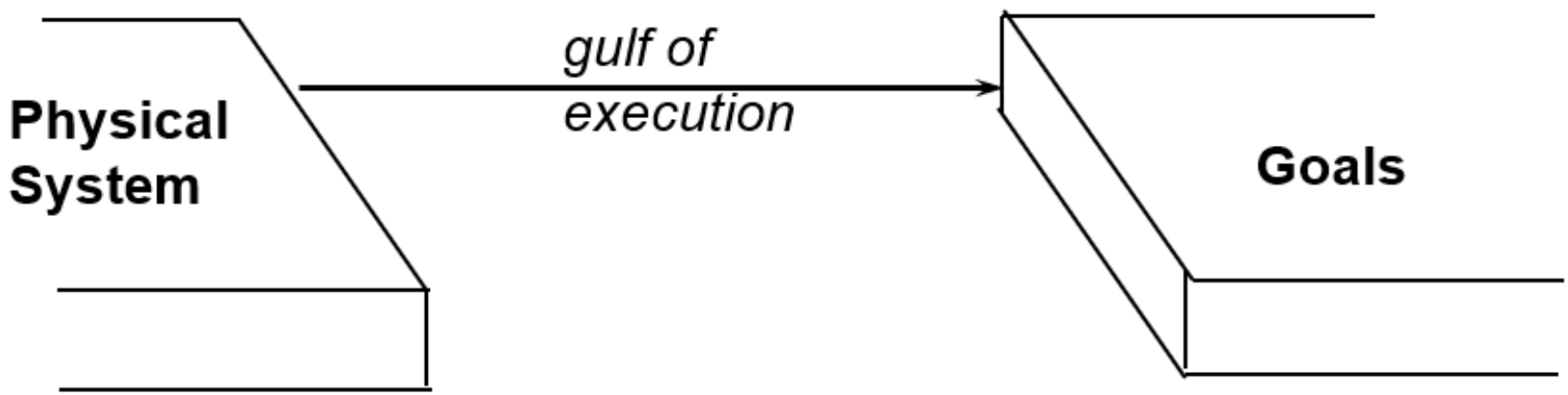
What the four stages model reveals

The "Gulf of Execution"

- Gulf: amount of effort exerted to transform intentions into selected and executed actions
- do actions provided by system correspond to the intentions of the user?
- 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)"

What the four stages model reveals

The "Gulf of Execution"



What the four stages model reveals

The "Gulf of Evaluation"

- Gulf: amount of effort exerted to interpret feedback
- can feedback be interpreted in terms of intentions and expectations?
- a good system:

  feedback easily interpreted as task expectations

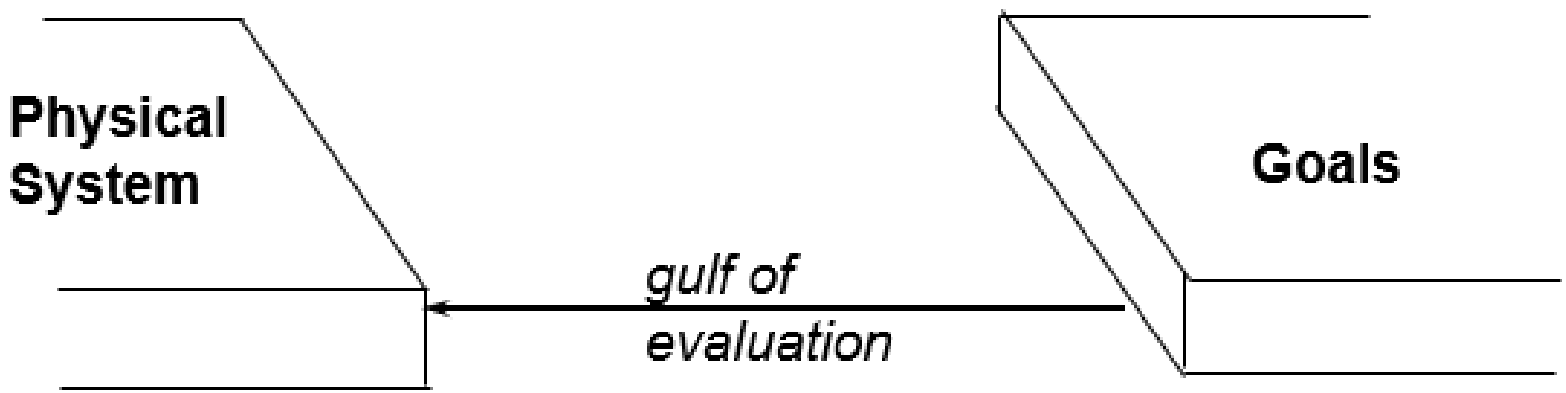
  - e.g. graphical simulation of text page being printed
- a bad system:

  no feedback or difficult to interpret feedback

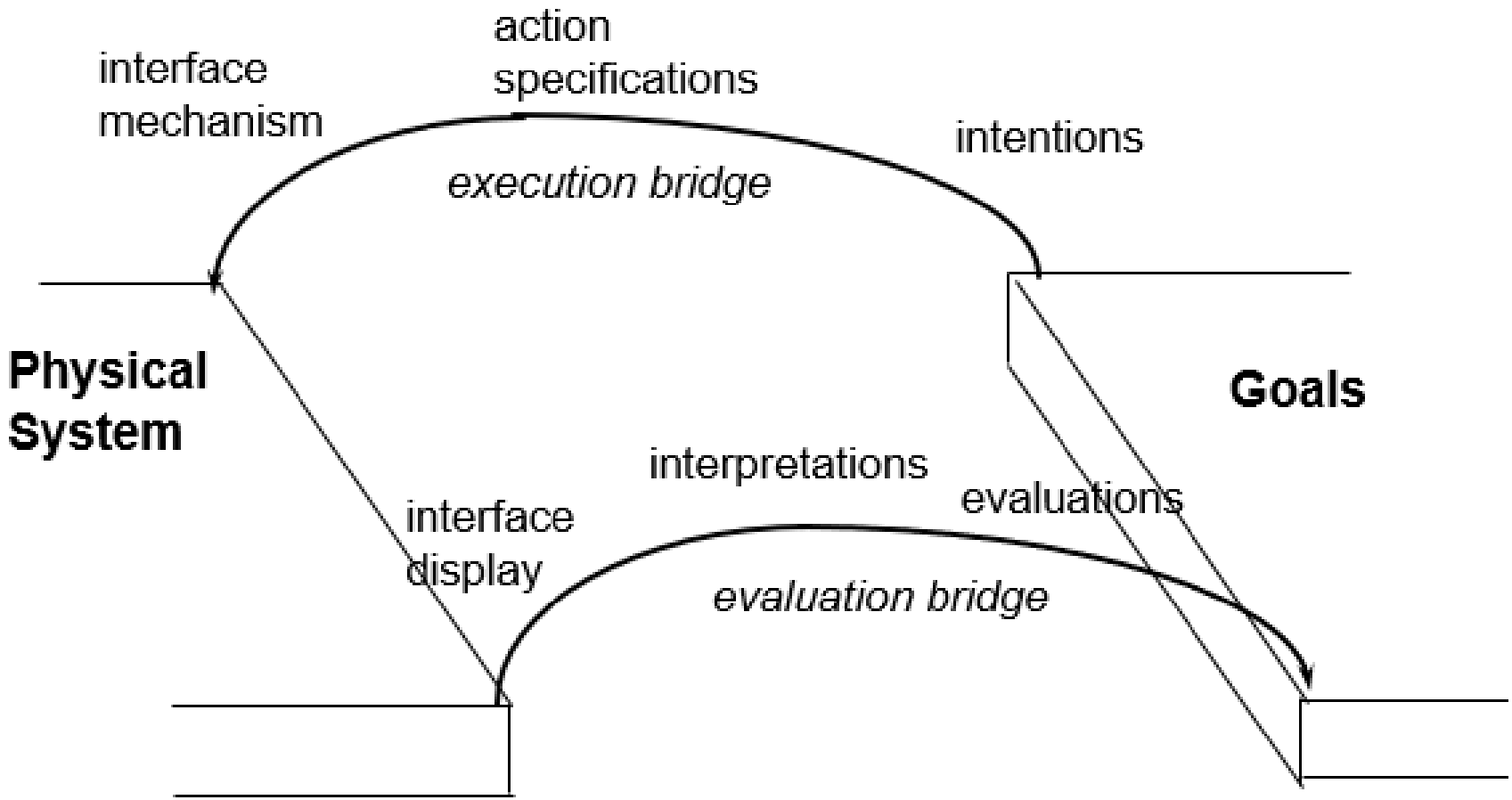
  - e.g. Unix: "$", "bus error", "command not found"

What the four stages model reveals

The "Gulf of Evaluation"



Bridging the Gulf of Execution and Evaluation



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?

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

Using four stages to ask design questions

Questions similar to principles of good design:

- Principle of transparency:
  - "the user is able to apply intellect directly to the task; the tool itself seems to disappear"

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

*Bibliography

- Saul Greenberg, **Understanding users and their tasks. High level models of human-computer behaviour**, University of Calgary, Canada
  http://pages.cpsc.ucalgary.ca/~saul/481/

- Mehmet Goktürk, **Fitts's Law**, in The Glossary of Human Computer Interaction
  https://www.interaction-design.org/literature/book/the-glossary-of-human-computer-interaction/fitts-s-law

- Zbynek Sterba, **Keystroke Level Modeling as a Method for Usability Quantification**
  https://medium.com/@zbynekste/keystroke-level-modeling-as-a-method-for-usability-quan