Programming the Interface

LECTURE NOTES

Design

- Conceptual: requirements, system expectation, needed information
- Physical: how to achieve objectives

Requirements

- Functional: what the interface must do
- Data: what needs to be available for processing
- Usability: user performance and satisfaction

System Models

- Dataflow: data the passes between processes
    - rectangle: source or destination of data
    - circle: process which transforms data
    - named link: transacted data
    - bucket: database or store
- Entity Relationship (ER)
    - entities: aggregate of data elements with a meaning
    - attributes: specific types of data
    - relationships: connections between entities

User Interface "Programming" Tools

- command processors, scripting languages (SQL, UNIX shell, HTML)
- menu systems (Mac, Windows)
- form fill-in systems (Netscape, databases)
- user interface toolkits (SUIT, NeXTStep, Visual Basic)
- window managers (spreadsheets, MacOS, Win95)
- user interface management systems (CLIM, JAVA)

Decision Types

- structural: end user's conceptual model
- functional: user actions and operations
- dialog: content and sequence of information exchange
    - semantics, units of meaning
    - messages, units of content
    - sequences, flow of content
- presentation: interaction objects and processes (widgets)
- pragmatic: use of hardware and physical space

Desirable Properties of a Conceptual and Implementation Model

sufficiency: all the needed information
necessity: only the needed information
understandability: easy to learn, easy to use
independence: modify constructs with minimal interaction
reusability: generic and general
consistency: same activity in same manner
minimality: no overlapping definitions and actions
orthogonality: each object accomplishes a different objective
compatibility: all models use similar concepts
implementability: easy to build

Usability requirements

learnability: time and effort to reach a level of proficiency
throughput: speed of execution and number of errors
flexibility: accommodation to changes in task and environment
attitude: satisfaction and acceptance

Task analysis techniques

Goals, tasks, actions
Hierarchical task analysis
Goals, operations, methods, selection rules (GOMS)
Task, semantic, syntactic, interaction

Usability testing techniques

direct observation
indirect observation (video recording)
verbal protocols (thinking aloud)
software logs
interviews (structured or flexible)
questionnaires
  checklist, rating, semantic differential, ranking

Potential measurement criteria

time to complete task
percentage of task completed
speed (percentage of task per unit time)
ratio of success to failure

time spent on errors
number of commands used
frequency of use of help or documentation
time spent using help
percentage of favorable or unfavorable user comments
number of repetitions of failed commands
number of runs of success or failure
number of times the interface misleads the user
number of good and bad features recalled by users
number of available commands not invoked
number of regressive behaviors
number fo users choosing or preferring system
number of times users have to work around a problem
number of times user is disrupted from task
number of times user loses control of system
number of times user expressed frustration or satisfaction