Computing Science GLASGOW

HCI4 06

GIST
GLASGOW INTERACTIVE SYSTEMS GROUP

Task Modelling and Analysis

Phil Gray

Overview

- Describing Tasks
- Predictive Task Models: GOMS
- Other Task Models
  - » UAN
    - – Cognitive Walkthrough
  - » Concur Task Trees
- Literate Development

Describing Tasks

- task description represents, more or less formally, selected aspects of interaction with a computer system
- may include
  - » user cognitive activity and state
  - » logical structure of tasks
  - » sequences of input actions
  - » display state
  - » representation of data and actions
  - » task execution context
- may be used for
  - » determining requirements
  - » identifying likely problems
  - » making predictions about user performance
  - » communicating external specification of design

Predictive Models: GOMS

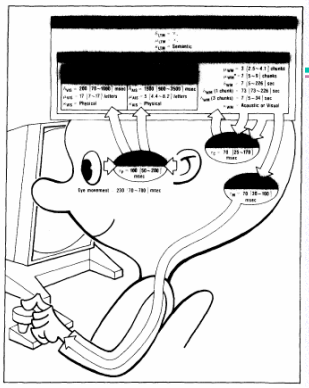
- User knowledge required to perform tasks is divided into:
  - » – GOALS
    - – knowledge of state which will successfully complete task
  - » – OPERATORS
    - – primitive motor and cognitive actions
  - » – METHODS
    - – combinations of operators to accomplish a goal
  - » – SELECTION RULES
    - – rules which determine the method to use when a choice is available
- Task analysis in GOMS (and most other methods) consists of successive decomposition of goals into subgoals which result in activation of selection rules and methods

GOMS: Levels of Analysis

• unit-task level	operators are complete tasks
• functional level	operators are system functions
• argument level	operators are components of function specification; e.g., command name and arguments
• keystroke level	operators are keystrokes, mouse movement, etc.; tasks can be given precise predictive completion times

GOMS Example

```
Goal: Edit-Unit-Task ... repeat until no more tasks
    Goal: Acquire-Unit-Task
        Goal: Turn-Page* ... if at end of page
        Goal: Get-From-Manuscript
    Goal: Execute-Unit-Task ...if edit task found
        Goal: Locate-Line ...if task not on current line
                Choose-Command
                [ select    Goal: Use-QS-Method
                                Goal: Specify-Command
                                Goal: Specify-Arg
                            Goal: Use-LF-Method
                                Goal: Specify-Command ] ... repeat until at line
                Goal: Verify-Loc
        Goal: Modify-Text
                Choose-Command
                [ select    Goal: Use S-Command
                                Goal: Specify-Command
                                Goal: Specify-Arg
                                Goal: Specify-Arg
                            Goal: Use-M-Command
                                Goal: Specify-Command
                                Goal: Specify-Command ...repeat until at text
                                Goal: Specify-Arg
                                Goal: Specify-Command]
        Goal: Verify Edit
```



Assessing Task "Cost" in GOMS

- STM load represented by "goal stack"
- LTM load represented by the number and complexity of the methods and selection rules
- if times can be given to execution of operators, then time to complete task can be predicted

GOMS Keystroke Time Parameters

- keystroke
  - 280 msec for average typist
  - 80 msec for best typist
  - 1200 msec for worst typist
- mental operator
  - time to retrieve chunk of information from LTM
  - 1.35 s
- pointing
  - average 1.1 sec
  - depends on Fitt's Law
- moving hands from keyboard to mouse
  - 400 msec
- literature varies somewhat in values given

pointing devices: Fitt's Law

- **$T = c1 + c2 \log_2 (D/S+0.5)$**
  where
  - **T = time to position the mouse (in seconds)**
  - **D = the distance to the object (in pixels)**
  - **S = the width of the object**
- **the constants for a mouse**
  c1 = 1.03 and c2 = 0.96
- **roughly the same as for the hand alone**

the limiting case: at the edge of the "hit area"

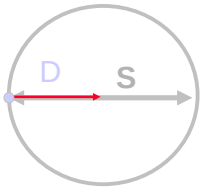
assume D = 5 and S = 10

$T = c1 + c2 \log_2 (5/10+ 0.5)$

$= c1 + c2 \log_2(1)$

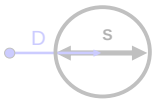
$= c1 + c2(0)$

$= c1$



increasing the distance to target...

D	S	$\log_2(2D/S)$	T
5	10	0	c_1
10	10	1	$c_1 + c_2$
20	10	2	$c_1 + 2c_2$
40	10	3	$c_1 + 3c_2$
80	10	4	$c_1 + 4c_2$



GOMS: Cons

- applies only to skilled users
- simplistic account of cognitive activity
- not good for highly parallel activity
- doesn't account for
  - learning
  - errors
  - mental workload
  - fatigue
  - individual differences
  - user attitudes
  - effects of the environment

Adding Temporal Information to the Task Structure

- Lots of choices
  - annotated task trees (ConcurTaskTrees)
  - task tables (UAN and XUAN)
  - state transition diagrams (including Petri Nets)
  - timeline-based representation
- Use the notation that suits the specification job

The User Action Notation

- UAN = User Action Notation
- semi-formal notation for describing the behaviour of user and system during human-computer interaction
- four descriptive components
  - user actions
  - user interface feedback
  - user interface state
  - application operations

The UAN (cont'd)

- additional features
  - special notation for certain types of user action & feedback
    - designed for direct manipulation style
  - may be augmented by
    - task trees
    - state transition diagrams
    - scenarios
    - other annotations (e.g., screen sketches or dumps)

UAN Tabular Representation

User Actions	System Feedback	UI State	Application Operations
select file icon	file icon is highlighted	currently selected object = file	
move cursor to **File** menu	cursor tracks mouse movement		
depress mouse button	**File** menu drops down		
move cursor over **Duplicate** menu item	**Duplicate** menu item is highlighted		
release mouse button	**File** menu disappears; "Copying" alert box appears while file is copied; icon labelled "Copy of <filename>" appears in same window as file icon		new file is created called "Copy of <filename>" containing duplicate of contents of file

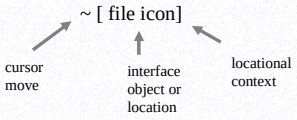
Note: columns may be omitted if empty

Why Use a Table?

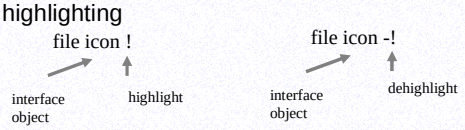
- "task trees" are an alternative
  - ConcurTaskTrees
  - clear view of the temporal relationships among subtasks
- tables better for
  - showing action-feedback exchanges
  - seeing fine-grained interaction problems

Notational Shortcuts: User Actions & Feedback

- cursor movement



- highlighting



UAN Subtasks

- user action column may include references to
  - » primitive actions
  - » subtasks
- each subtask must be defined in a separate UAN task table

Task: create document
launch application ~ [FILE] M v ~ [NEW] M ^

subtask - define elsewhere

Parameterised Tasks

Task: execute (button)

User Actions	System Feedback	Application Operations
~ [button]	cursor tracks	
M v	button !	
M ^	button - !	execute button action

Task: open (file_name)

User Actions	System Feedback	Application Operations
file_name not visible: (scrollup | scrolldown)*	list scrolls up or down	
~[file_name]		
M ^v	file_name !	
execute (OK)		file is opened

Temporal Relationships Among Tasks

sequence	**A B**
repetition	**A***
conditionality	**c : A**
optionality	**{A}**
choice	**A | B**
repeating choice	**(A | B)***
order independence	**A & B**
interruptibility	**A -> B**
interleavability	**A <-> B**
concurrency	**A || B**
waiting	**A (t > n)**

UAN: Pros & Cons

- advantages
  - » low-cost method
    - » easy to learn
    - » extensible
  - » can be understood by users and designers
  - » well-suited to direct manipulation
- but
  - » omits reference to goals
  - » task hierarchy hard to discern
  - » can be hard to identify information flows
  - » semantics are not rigorously defined

Things to Do with UAN

- design communication
- analytic evaluation
  - » UAN-based cognitive walkthrough
- literate development
  - » Clarke's LD system
  - » Graham's Clock method

Analytic Evaluation

- analysing for errors
  - » informal heuristic analysis
  - » requires
    - – ideal description
    - – actual description
- cognitive walkthrough
  - » structured heuristic analysis
  - » requires
    - – scenario
    - – task-oriented scenario description
    - – set of walkthrough questions

UAN-based Cognitive Walkthrough Method

- create a scenario in terms of a main task, subtasks, actions and system feedback
- perform the walkthrough by going through the goals and actions, and answering a set of questions
- record any problems found

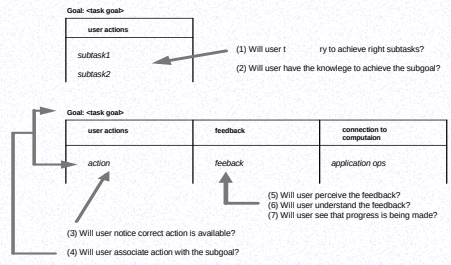
The Questions

- goals
  - » will the user try to achieve the correct goal?
  - » does the user have the knowledge to achieve the goal?
- actions
  - » will the user notice the correct action is available?
  - » will the user associate the action with the goal?

The Questions (cont'd)

- Feedback
  - » will the user perceive the feedback?
  - » will the user understand the feedback?
  - » will the user know that progress towards the goal is being made?

UAN-CW Questions



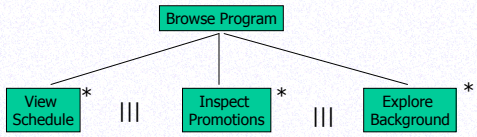
Concur Task Trees

- Notation emphasising
  - » Hierarchical structure
  - » Graphical syntax
  - » Concurrency
- Linked to UML
- Supported by editing and analysis tools

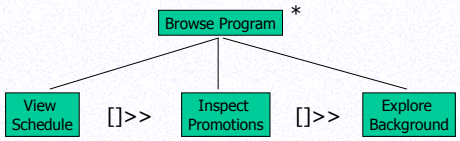
ConcurTaskTree operators

T [] T2	choice
T1 >> T2	enabling
T1 ||| T2	interleaving
T1 |[]| T2	synchronization
T1 []>> T2	enabling with info passing
T1 [> T2	deactivation
T*	iteration
T(n)	finite iteration
[T]	optional task

ConcurTaskTrees Example



ConcurTaskTrees Example



CTTE

- http://giove.cnuce.cnr.it/ctte.html
- Tool for supporting the development and analysis of CTTs
- Mori, G; Paterno, F; Santoro, C. CTTE: Support for Developing and Analyzing Task Models for Interactive Systems Design. IEEE Trans Soft Eng, 28 (8), 2002, pp. 797-813

Things to do with UAN

Literate Development

- an approach to system development, not a method
- emphasises links among different types of design information
- supports
  - » better understanding of design
  - » use of analysis information throughout development process
  - » evaluation of consequences of design changes

Design Representations

Role	Notations
contextual	affinity diagram, domain model
environment	scenarios, screen sketches
behavioural	UAN, CTT
constructional	UML, Petri Nets
design rationale	QOC, IBIS

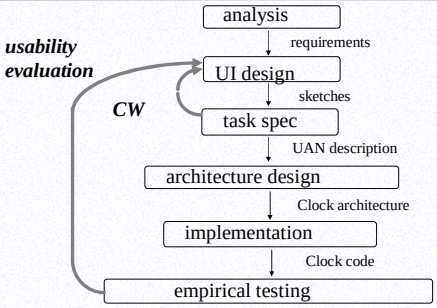
Clarke's LD System

- experiment in computer-support for managing and using contextual information
- supports creating and linking of
  - » checklist-based contextual items
  - » scenarios
  - » UAN task descriptions
  - » QOC design rationales
  - » NUF system specification

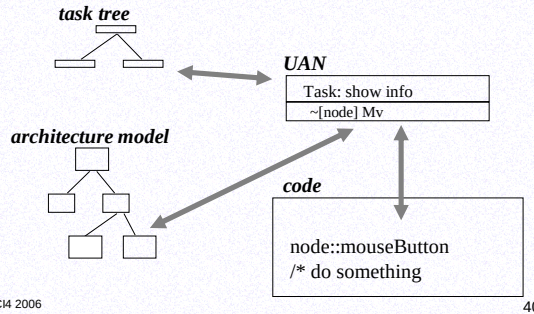
LD Evaluation

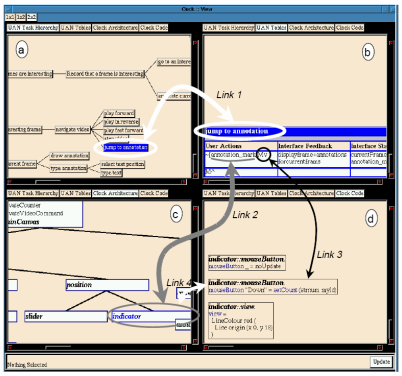
- informal evaluation
  - » links were judged useful
  - » variety of uses of information
  - » subjects wanted
    - – weighted relationships
    - – ways of finding missing relationships
    - – graphical description of UI as design representation
  - » quality of tool interface is important

The Clock Method



Linking the Representations





Model-Driven Development

- The vision: to be able to generate & evaluate a system from a set of models that together specify the
  - » Domain
  - » Tasks
  - » Context of Use
  - » Interaction techniques
- USIXML and its toolset is an attempt to achieve this