

Task-Centered User Interface Design

CMPT 281

Outline

- What is TCUID?
- TCUID process
- Finding and working with users
- Developing task examples
- Evaluation with task-centered walkthrough
- Example: the Cheap Shop interface

What is TCUID?

- A design process
 - one type of user-centred design
 - structured around specific tasks that users want to accomplish with the system
 - emphasis on early consideration of tasks in design
 - emphasis on concrete detail
- Tasks are used to:
 - aid in making design decisions
 - evaluate the design as it is developed

The TCUID process

- figure out who's going to use the system to do what
- choose representative tasks for task-centered design
- borrow
- rough out a design
- think about it
- create a mock-up or prototype
- test it with users
- iterate
- build it
- track it
- change it

The TCUID process

- figure out who's going to use the system to do what
- choose representative tasks for task-centered design
- borrow (look for inspiration)
- rough out a design (paper prototype)
- think about it (walkthrough)
- create a mock-up or prototype
- test it with users
- iterate
- build it
- track it
- change it

Finding potential users

- Who would use the system to do what?
 - *Exactly* who, and *exactly* what?
- Find potential users of your system
 - Real people! - not managers or other designers
 - If you can't find them, who will actually buy/use your system?
 - Consider typical and atypical users
- Find out how the system would fit with their work
 - Are they interested? Why or why not?
- Learn about the user's tasks
 - Develop concrete, detailed examples of tasks they perform or want to perform that your system should support

Choose representative tasks

- What are the core tasks the system must support?
- Consider *importance* and *frequency*
- Start with 3-5 core tasks
 - check your decisions with users
- Write out task descriptions for these tasks

Task descriptions

- “Change the speed limit on Canyon Boulevard eastbound between Arapahoe and 9th. Calculate projected traffic flows on Arapahoe west of 6th assuming Canyon speeds between 25 and 55 in increments of 5 mph.”
- Read Chapters 0,1,2, 4.1:
 - **Task-Centered User Interface Design: A Practical Introduction**
 - <http://hcibib.org/tcuid/>

Task descriptions

- Says what the user wants to do but not how they would do it
 - no assumptions made about the interface
 - can compare different design alternatives in a fair way
- Are very specific
 - says exactly what the user wants to do
 - specifies actual items the user would eventually want to enter, identify, or manipulate
 - states constraints or goals in task completion
 - e.g. time: as fast as possible; less than 15 seconds
 - e.g. quality: how much room for error?

Task descriptions

- Describe a complete job
 - not just a list of simple things the system should do
 - does more than present sub-goals
 - forces designer to consider how features will work together
 - considers the flow of information over a time period
 - where does information come from?
 - where does it go?
 - what has to happen next?
- What is a complete job?
 - consider temporal and spatial breaks
 - check your decisions with users

Task descriptions

- Say who the users are
 - be specific – important details are easy to overlook
 - design success is strongly influenced by what users know
 - reflect the real interests of real users
- Are evaluated
 - Circulate descriptions to users, and rewrite if needed
 - ask users for:
 - omissions
 - corrections
 - clarifications
 - suggestions
 - Remember users aren't always right
 - conflict can tell you that you are missing something

Borrow ideas

- Tough to be beat existing interfaces
- Helps learning
- Saves time in designing and building
- Don't actually steal, just look for inspiration

Borrow ideas

- How to actually design the UI?
 - Look for existing interfaces that work
 - Build ideas from those interfaces into your systems
 - e.g. high-level interaction paradigms
 - e.g. low-level control/display decisions
 - Work within existing interface frameworks
 - e.g., GUI toolkits
 - make use of existing applications

Rough out a design

- Low-fidelity prototyping
 - paper & pencil
 - just draw the interface on paper
 - don't worry about consistency or aesthetics
- Focus on the core tasks
 - don't add extra functionality
 - make sure the prototype covers all of the tasks

Think about it (= evaluate)

- Evaluate your rough design
 - in terms of the tasks you were trying to support
 - without a user!
 - can catch errors that testers wouldn't catch
- Could use existing evaluation methods:
 - Cognitive Walkthrough
 - Heuristic Evaluation

Cognitive Walkthrough

- Cognitive walkthrough
 - a UI evaluation technique where designers step through a task and attempt to complete the task with a prototype
 - “simulated realistic user” based on the detailed user description
 - goal is to identify obvious problems with the design
- Stages:
 - create ‘runnable’ task scenarios
 - do a walkthrough for each scenario
- Reading: “Performing a cognitive walkthrough”:
www.cc.gatech.edu/classes/cs3302/documents/cog.walk.html

Task Scenarios

- All the information needed to test the design
 - all task information
 - detailed sketches of all needed screens
- Create a scenario for each task
 - design specific
 - what the user would have to do to carry out the task with this particular design
 - what the user would see step-by-step and at key moments in performing the task
- Forces us to:
 - get specific about our design
 - consider how the various features work together to accomplish real work

Walkthroughs

- Select one of the task scenarios
- For each user step or action in the task:
 - Carry out the action with the interface
 - Ask:
 - would the user have done this action in this way?
 - would the user's expected knowledge be sufficient
 - would the user know what to do, be able to find control/command, and know that the control would have the desired effect?
 - would the action be possible given the user's situation?
 - is the feedback understandable and sufficient?
 - if no:
 - you've located a problem in the interface
 - once a problem is identified, assume it has been repaired
 - go to the next step in the task

Example: Cheap Shop

- *At Cheap Shop, people browse a catalog and then order goods from a clerk.*
- Task example 1:
 - A man caring for a demanding toddler buys an umbrella stroller (red is preferred, but blue is acceptable), pays for it in cash, and uses it immediately.
- Task example 2:
 - An elderly arthritic woman is comparing prices for a child's bedroom set, consisting of a wooden desk, a chair, a single bed, a mattress, a bedspread, and a pillow. She takes the description and total cost away with her, to check against other stores. Two hours later, she returns and decides to buy everything but the chair.

Example: Cheap Shop

- Task example 3:
 - A “Cheap Shop” clerk, who is the sole salesperson in the store, is given a list of 10 items by a customer who does not want to use the computer.
 - 4 pine chairs, 1 pine table, 6 blue place mats, 6 “duo” forks, 6 “duo” table spoons, 6 “duo” teaspoons, 6 “duo” knives, 1 “tot” tricycle, 1 red ball, 1 “silva” croquet set
 - After seeing the total, the customer decides to take all but the silverware, and then adds 1 blue ball to the list.
 - The customer then changes their mind about paying by credit card, and decides to pay cash. The customer wants the items delivered to his home the day after tomorrow.
 - While this is occurring, 6 other customers are waiting for the salesperson.

Screen 1

Cheap Shop Catalog Store ☐ ☐ ✕
Danderly software, screen A1.1

Purchaser

Name: Phone:

Postal Code: Province: City:

Delivery Address:

Today's date:

Credit Card No.: for dept use: validation id:

Catalog Item

Number: Quantity: Cost/item: Total:

Balance Owing:

Next Catalog Item (PF5)

Trigger Invoice (PF8)

Screen 2

Cheap Shop Catalog Store ☐ ☐ ✕
Danderly software, screen A1.2

Catalog Item

Number: Quantity: Cost/item: Total:

Balance Owing:

Next Catalog Item (PF8)

Trigger Invoice (PF5)

Prototype specifications

- Basic usage:
 - Shoppers enter item codes from a catalog onto the screen.
- To create an order
 - screen 1: shopper enters personal information and first order
 - text is entered via keyboard
 - the tab key or mouse is used to go between fields.
- Further orders
 - shoppers go to screen 2 by pressing the 'Next Catalog Item' button
- Order completion
 - shoppers select 'Trigger Invoice'.
 - the system sends the order to shipping and billing
 - the system returns to a blank screen #1

Prototype specifications

- To cancel an order
 - Shoppers do not enter any input for 30 seconds (as if they walk away)
 - The system will then clear all fields and return to the main screen
- Input checking
 - all input fields checked when either button is pressed.
 - erroneous fields will blink for 3 seconds, and will then be cleared.
 - the shopper can then re-enter the correct values in those fields.