ELEC 372

Engineering Software Structures

Week 11: Usability Engineering
Lecture 1

Usability as a Quality Attribute

- "Usability" of an application refers to its
  - Ease of use
  - Ease of learning
  - Appropriateness to the task being performed
- Usability can be increased through
  - Early evaluation of the user interface
  - Extensive and iterative requirements analysis
  - Use of design guidelines and usability testing
  - User and task modeling
- Usability engineering integrates these activities in the software development process

Disciplines Contributing to Usability Engineering

- User interface design, evaluation is multidisciplinary
  - Graphic designers: appearance of displays, icons
  - UI designers: interaction design
  - Psychologists: understanding user's capabilities
  - Sociologists: understanding work context in which application will be deployed
  - Software engineers: implementation

Examples of Usability Problems

- System is poorly documented
- System is hard to use
- System is hard to learn
- System doesn't match needs of users
- System is buggy or poorly supported
- …etc.

System Hard to Learn



- Telephone on my desk
  - E.g., buttons for "Ring Again", "Redial"
  - Voice mail
    - listen to message = 2
    - delete message = 76
    - save message = ???
    - incomplete prompts

System Hard to Use



- Web Banking System
  - Different screens for
    - Balance enquiry
    - Transfer funds
    - Pay bills
    - Short, long term history
  - E.g., paying visa requires navigating > 10 screens

System Untrusted



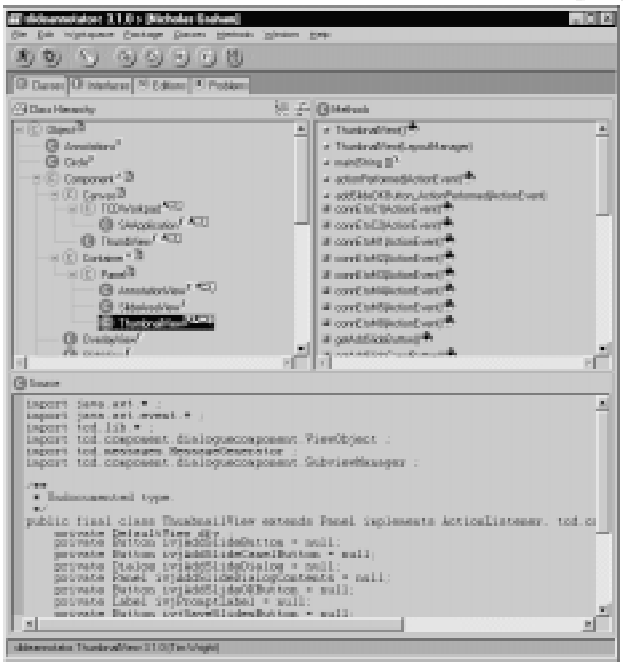
- E-bay online auction
  - Worry about safety of credit card use
  - Worry about honesty of vendors

System is Buggy, Poorly-Supported



- Positioning pictures in MS-Word
  - Positioning sometimes random, hard to tune
  - Updating cross-references repositions figures!

System Doesn't Match Needs of Users



- IBM VisualAge for Java in University Environment
  - Each machine can support only one user workspace
  - Incompatible with "walk up" terminal use in student labs

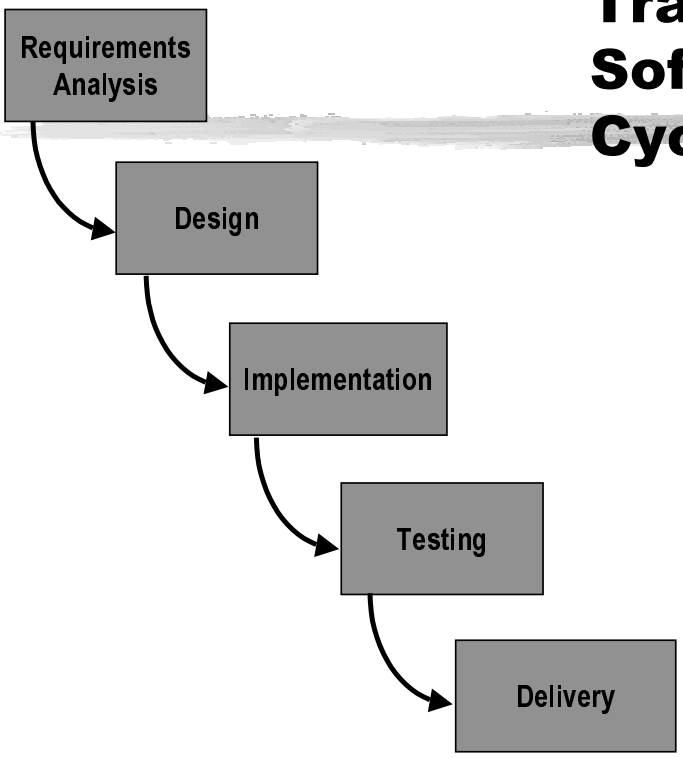
Development of Interactive Systems

- 1960's: Monolithic mainframes, cards
  - Expensive, batch, select user group
- 1970's: Glass teletypes, mini-computers
  - Cheaper, interactive, somewhat select
- 1980's: Bitmapped displays, novel I/O
  - Cheap, highly interactive, available
- 1990's: GUI's, networking
  - Extremely cheap, connected, ubiquitous
- 2000's: Information appliances, wireless
  - Extremely cheap, connected, mobile

Question for System Developers

- Can we develop systems that
  - fit the organization?
  - fit the user?
  - still meet time, budget constraints of project?

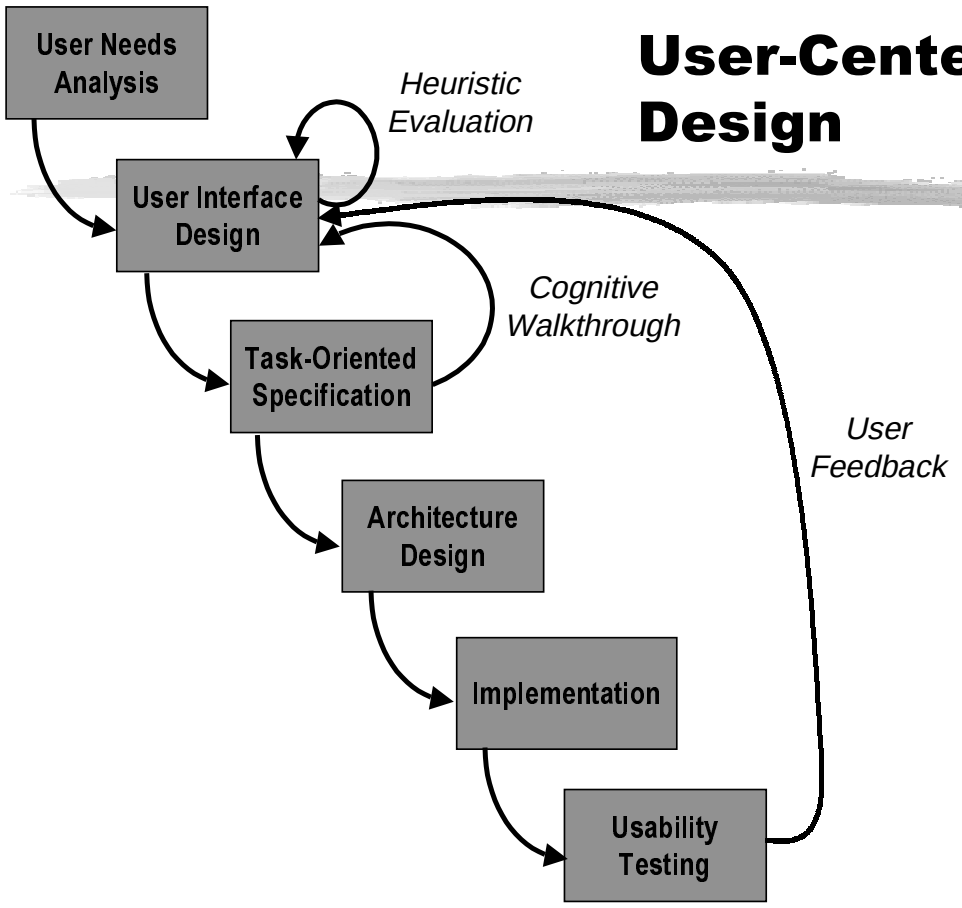
Traditional Software Life Cycle



Problems with Traditional Life Cycle

- Usability problems caught late
  - Expensive to repair when found
  - Developers become attached to designs
- If project running late, usability may be sacrificed

User-Centered Design



Goal of User-Centered Design

- Involvement and consideration of target users throughout process
- Continuous evaluation of design
- Detection of problems early

User Needs Analysis

- User characterization
- Modeling user activities, user tasks
- Modeling context of use

User Interface Design

- Creative process
  - Brainstorming, metaphor, scenarios, prototyping
- User participation
- Evaluation through heuristic analysis, user feedback

Task-Oriented Specification

- Description of how user tasks carried out with system
- Extension of task model
- Bridges gap between design and implementation
- Evaluation as cognitive walkthrough

Usability Testing

- Testing with members of user group
- Can be done with prototypes or real system

ELEC 372

Engineering Software Structures

Week 11: Usability Engineering

Lecture 2

User Interface Design Guidelines

- Capture design knowledge
- General guidelines capture high-level attributes that make systems more usable
- Specific guidelines
  - Enforce corporate or platform style
  - Ensure consistency within and across applications
- Guidelines may be
  - Validated by psychological research
  - Based on experience of expert designers

General UI Design Guidelines

[Baecker et al., Nielsen]

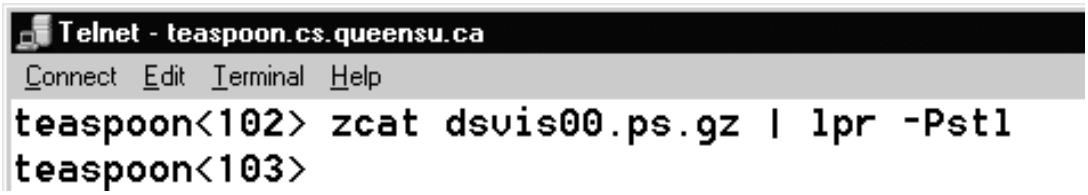
- Know the User
- Minimize Memorization
- Optimize Operations
- Engineer for Errors
- Avoid Modes
- Be Consistent

Minimize Memorization

- Don't require users to memorize complex commands
  - Make commands visible through menus, icons, palettes
- Don't require users to memorize information from one screen to the next
- Short term memory holds 7±2 chunks of information
  [George Miller, *The Psychology of Communication.* Basic Books, 1967]

Minimize Memorization

- Unix command to print a compressed file to the 'stl' printer



Minimize Memorization

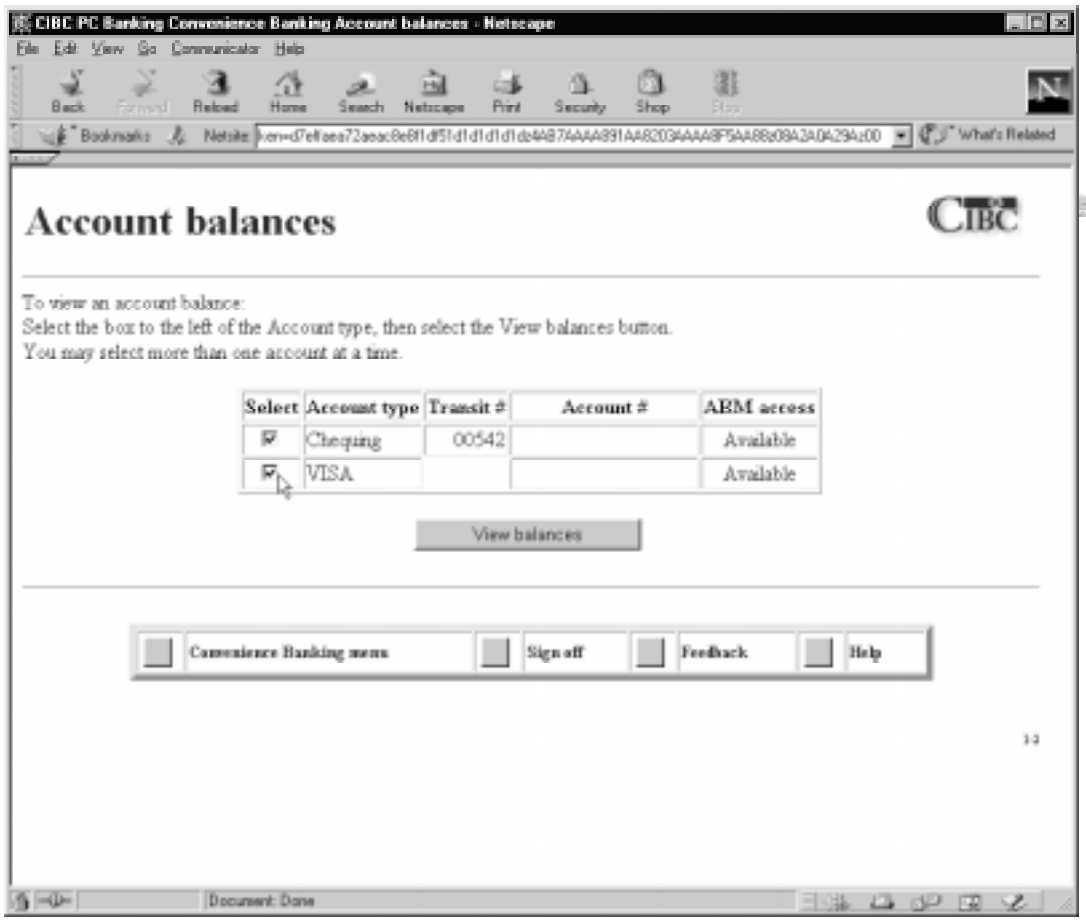


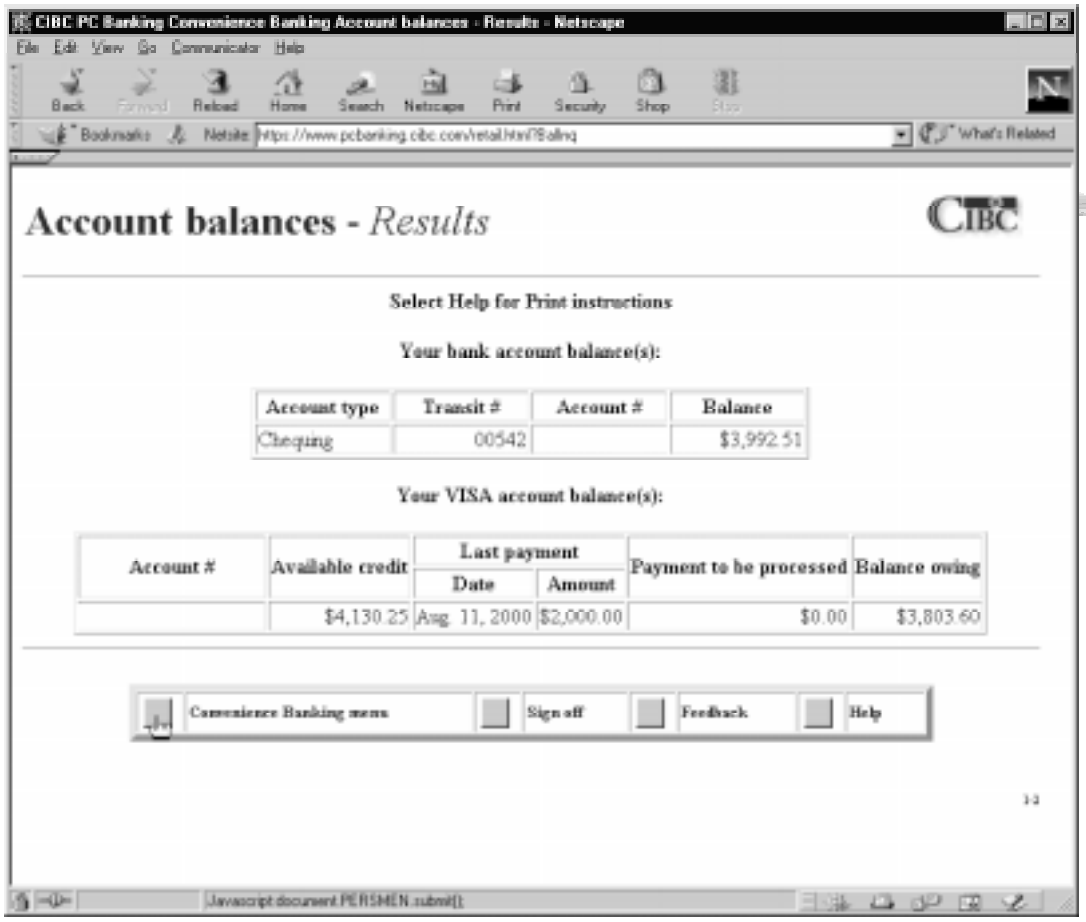
- Microsoft Word printing from menu

Minimize Memorization

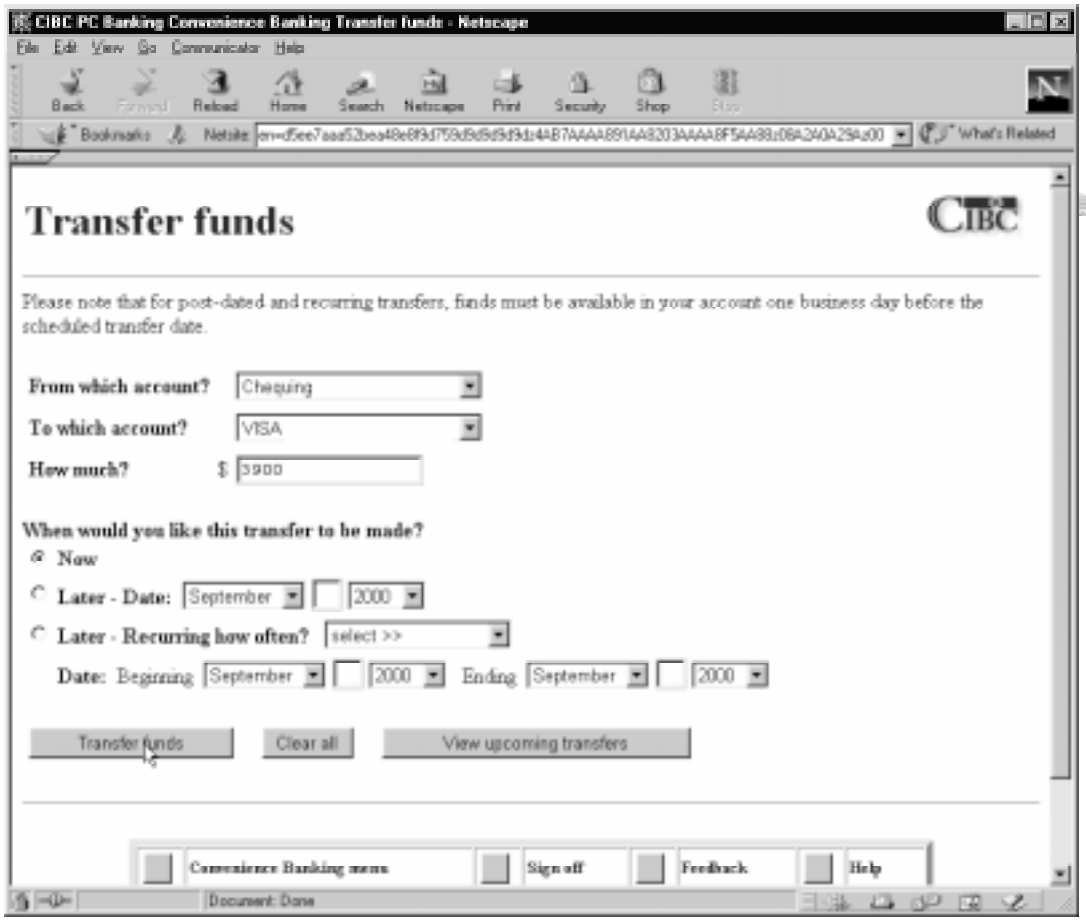
- Paying visa bill via online banking
  - Ensure user has available information required to carry out task

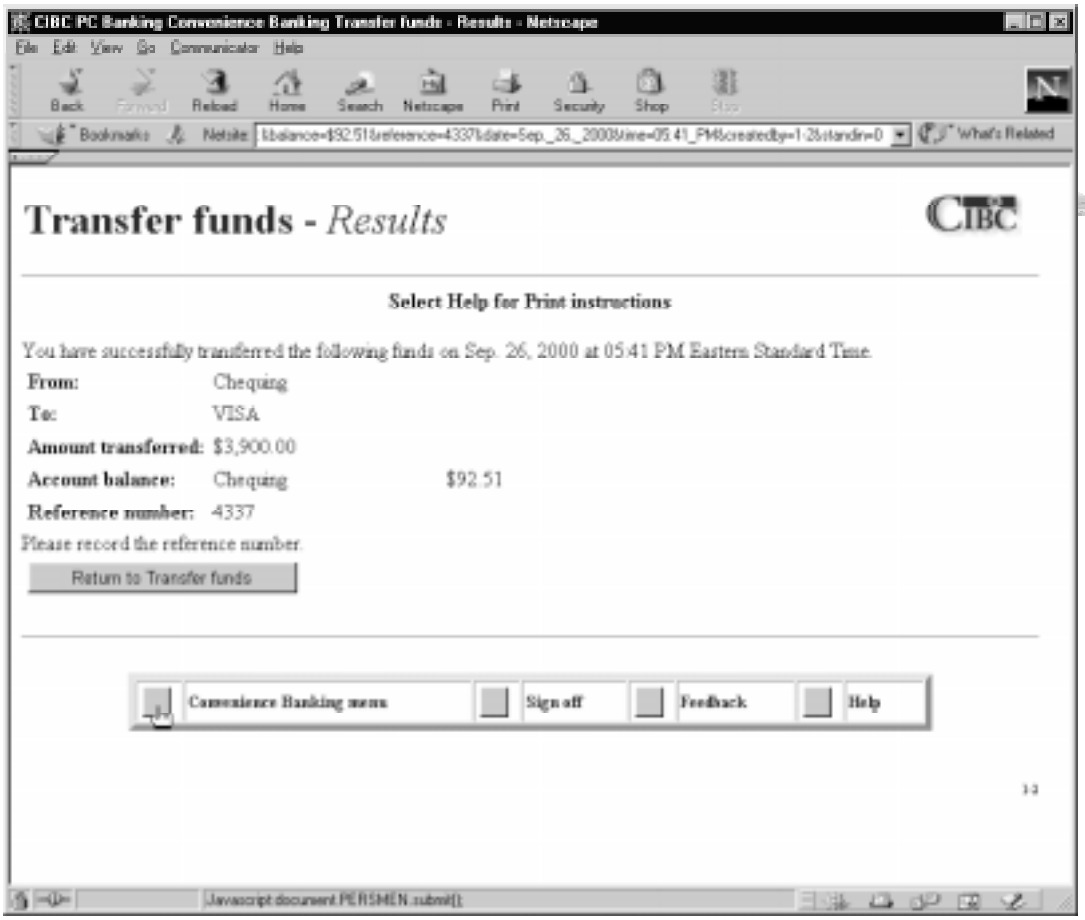

CIBC Convenience Banking - Netscape
Convenience Banking
CIBC
Account balances
Pay bills
Recent transactions
Transfer funds
Detailed account history
Session summary
Download transactions
Stop payments
Manage bank card
Sign off
View Legal Agreement
Feedback
Help


CIBC PC Banking Convenience Banking Account balances - Netscape
Account balances
CIBC
To view an account balance:
Select the box to the left of the Account type, then select the View balances button.
You may select more than one account at a time.
Select
Account type
Transit #
Account #
ABM access
Chequing
00542
Available
VISA
Available
View balances
Convenience Banking menu
Sign off
Feedback
Help
Document: Done


CIBC PC Banking Convenience Banking Account balances - Results - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Netsite: https://www.pcbanking.cibc.com/retail/html/?balinq
What's Related
Account balances - Results
CIBC
Select Help for Print instructions
Your bank account balance(s):
Account type
Transit #
Account #
Balance
Chequing
00542
$3,992.51
Your VISA account balance(s):
Account #
Available credit
Last payment
Date
Amount
Payment to be processed
Balance owing
$4,130.25
Aug 11, 2000
$2,000.00
$0.00
$3,803.60
Convenience Banking menu
Sign off
Feedback
Help
Javascript document.PERSMEN.submit();


CIBC Convenience Banking - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Netsite: https://www.pcbanking.cibc.com/retail/html/?Menu
What's Related
Convenience Banking
CIBC
Account balances
Pay bills
Recent transactions
Transfer funds
Detailed account history
Session summary
Download transactions
Stop payments
Manage bank card
Sign off
View Legal Agreement
Feedback
Help


CIBC PC Banking Convenience Banking Transfer funds - Netscape
Transfer funds
CIBC
Please note that for post-dated and recurring transfers, funds must be available in your account one business day before the scheduled transfer date.
From which account?
Chequing
To which account?
VISA
How much?
$ 3900
When would you like this transfer to be made?
Now
Later - Date:
September
2000
Later - Recurring how often?
select >>
Date: Beginning
September
2000
Ending
September
2000
Transfer funds
Clear all
View upcoming transfers
Convenience Banking menu
Sign off
Feedback
Help
Document: Done


CIBC PC Banking Convenience Banking Transfer funds - Results - Netscape
Transfer funds - Results
CIBC
Select Help for Print instructions
You have successfully transferred the following funds on Sep. 26, 2000 at 05:41 PM Eastern Standard Time.
From: Chequing
To: VISA
Amount transferred: $3,900.00
Account balance: Chequing $92.51
Reference number: 4337
Please record the reference number.
Return to Transfer funds
Convenience Banking menu
Sign off
Feedback
Help
Javascript document.PERSMEN.submit()

Optimize Operations

- Frequently done operations should be simple, quick, error-free
  - Interaction techniques: accelerators, macros
  - Dialogues: streamline for frequent operations
  - UI can adapt for expert users
    - Hard to design such adaptation so that it's not confusing

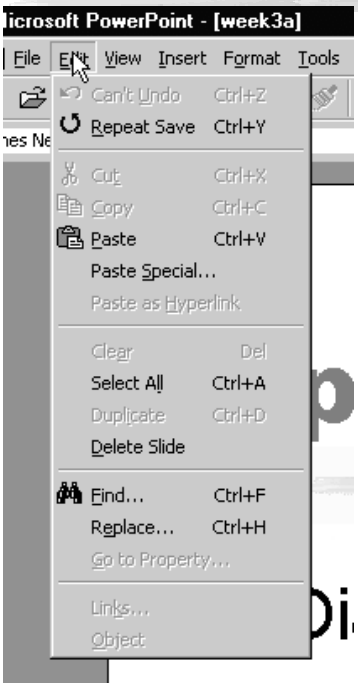
Optimize Operations

- Vi text editor
  - Command to change all old Queen's phone numbers into new phone numbers with area code
  - E.g., 545-6526 -> (613) 533-6526
  - Make sure we only do old Queen's phone numbers
    - 545-dddd
  - **`g/545-\([0-9][0-9][0-9][0-9]\)/s//(613) 533-\1/g`**

Optimize Operations

- Dialogue for paying Visa could be optimized
- How would you do this?

Optimize Operations



- *Keyboard accelerators* allow frequent operations to be performed quickly
- Showing them on the menu minimizes memorization

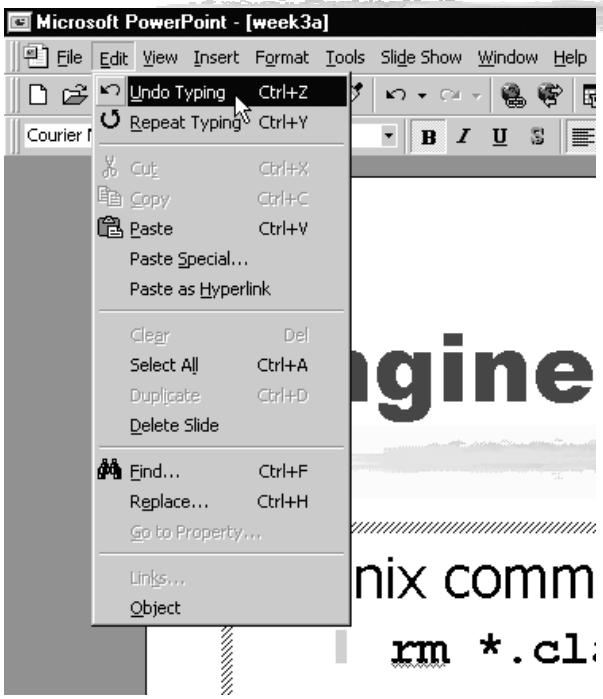
Engineer for Errors

- We tend to assume that correct usage is the norm
- Should treat erroneous use as the norm
  - People exploring, learning make mistakes
- Application should be designed so that mistakes are not costly
  - Decreases insecurity
  - Encourages exploration of system
- Errors may be one-time or recurring

Engineer for Errors

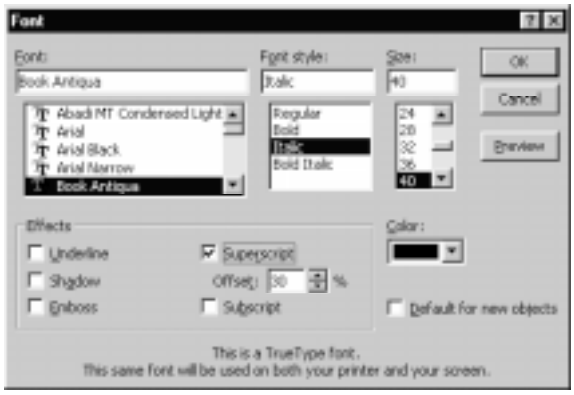
- Unix command
  - `rm *.class`
- removes all files ending with ".class" from directory
- However, Unix command
  - `rm * .class`
- removes *all* files from directory, and reports message
  - `.class: not found`

Engineer for Errors



- **Undo** operation supports exploration, allows errors to be undone
- Two variants -- assume UI starts in state s1, and user performs actions <a1> then <a2>, then selects undo twice
  - Undo-undo: second undo undoes first undo
    - s1 <a1> s2 <a2> s3 <undo> s2 <undo> s3
  - Undo-redo: continues undoing through history
    - s1 <a1> s2 <a2> s3 <undo> s2 <undo> s1

Engineer for Errors



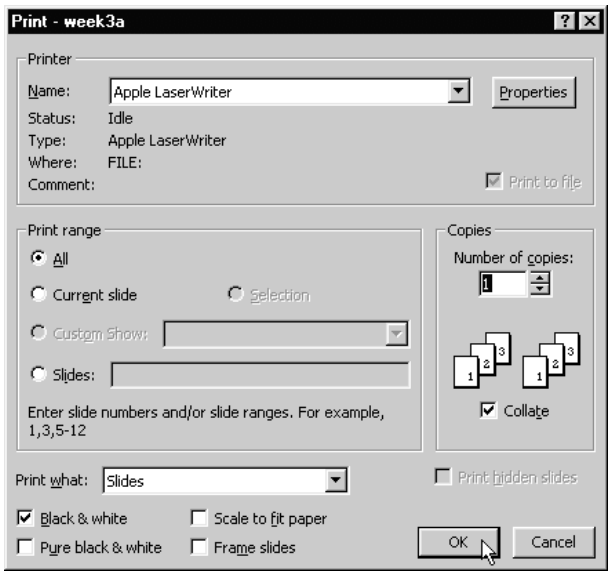
- "Cancel" operation in dialogue returns state of system to state before dialogue started

Engineer for Errors



- "lpr" command fails if:
  - "lpr" misspelled
  - name of printer unknown or misspelled
  - name of file misspelled

Engineer for Errors

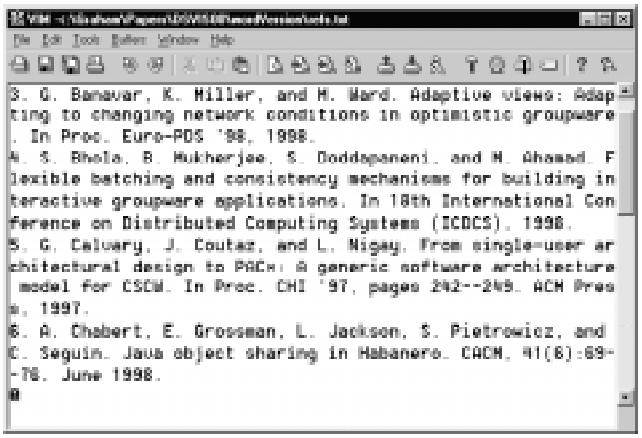


- MS Windows print dialogue box
  - Names of printers chosen from menu, therefore no error possible
- Errors still possible
  - Select page range is free form
  - If you don't want to print black and white, easy to miss check box that has to be reset

Avoid Modes

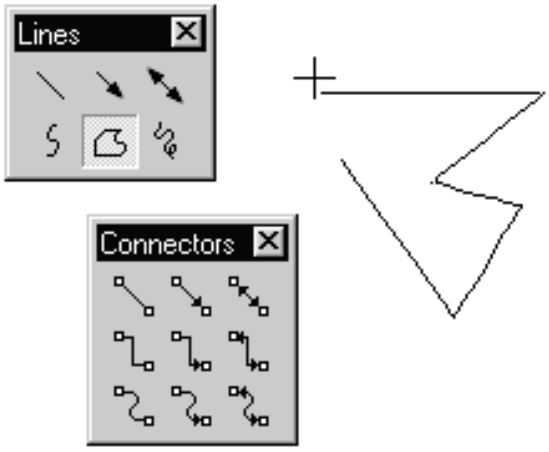
- Modes cause an interpretation on input depending on context
- Danger is
  - user won't understand what context they are in
  - user will forget what context they are in
  - imposes cognitive burden of thinking about "what does 'k' key mean now?"
- Modes can be acceptable if they are visible and intuitive

Avoid Modes



- In *vi* editor, there are
  - *Text mode:* typing enters text into document
    - e.g., typing "xxx" enters "xxx" at current insertion point
  - *Command mode:* characters interpreted as commands
    - e.g., typing "xxx" deletes three characters forward from insertion point
  - Default is command mode
    - Commands such as "i" (insert), "o" (add line) enter text mode
    - <Esc> key enters command mode
  - No way of seeing what mode you are in

Avoid Modes

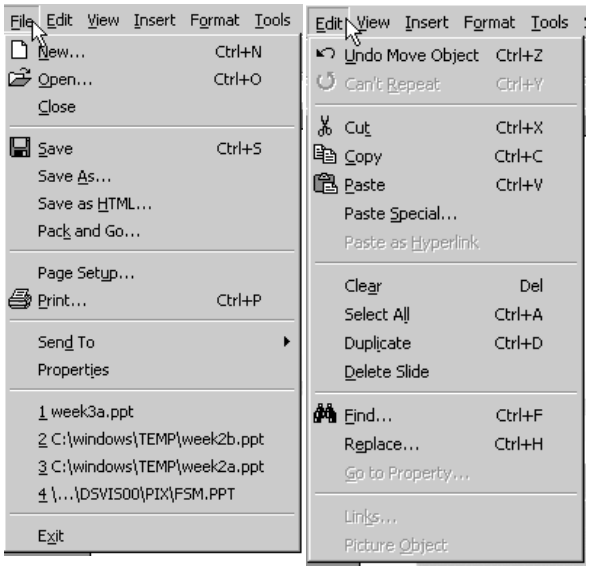


- Modes can be made visible
  - E.g., selecting from palette makes it obvious what modes available
  - Selection highlighted
  - Current mode shown in tracking symbol ("+")

Be Consistent

- Consistency allows users to reuse knowledge across and within applications
- Failure to be consistent can lead to errors, frustration

Be Consistent



- Follow conventions for menu layout
  - "File" has "New", "Open", "Close", "Save", "Save As"
  - "Edit" has "Undo", "Redo", "Cut", "Copy", "Paste"
- Follow conventions for keyboard accelerators
  - ^P - print
  - ^C - copy
  - ^X - cut
  - ^V - paste
  - ^A - select all
  - ^F - find

Be Consistent

- Use consistent date format
- Use consistent screen layout
- Use same navigation commands from screen to screen
- Use consistent colour (if colour matters)
- Use consistent dialogues
  - E.g., dialogues for printing, saving a file should always be the same

General UI Design Guidelines

[Baecker et al., Nielsen]

- Know the User
- Minimize Memorization
- Optimize Operations
- Engineer for Errors
- Avoid Modes
- Be Consistent

Detailed Design Guidelines

[Helander, ed., Handbook of HCI]

- Use of Text
- Placing and Sequencing of Information
- Presenting Text
- Graphics

Use of Text

- How much information?
  - 25%-60% of display should be used
- Reducing information
  - Make appropriate use of abbreviations
  - Avoid unnecessary detail
  - Use concise wording
  - Use familiar data formats
  - Use tabular formats with headings

Use of Text

- Grouping information
  - Highlighting relationships among data
  - Can use
    - Format
    - Colours
    - Outlining
    - Highlighting
    - Font

Use of Text

Cisc 124 Grades

John Smith received a grade of 47

Jen Wu received a grade of 92

Mike Jenkins received a grade of 86

Nick Graham received a grade of 63

- Hard to extract information from this format
  - What grade did a particular person get?
  - What is highest/lowest grade?
  - How many people failed?

Use of Text

Cisc 124 Grades	
John Smith	**47**
Jen Wu	**92**
Mike Jenkins	**86**
Nick Graham	**63**

Use of Text

Cisc 124 Grades	
John Smith	**47**
Jen Wu	**92**
Mike Jenkins	**86**
Nick Graham	**63**

- Tabular format eases reading
- Removed Unnecessary Detail
- Colour and font emphasize vertical relationship
  - Group names together and group grades together
- Multiple fonts, text colour, underlining are ugly and don't add any information
- Insufficient contrast between colours in title

Use of Text

Cisc 124 Grades	
Nick Graham	63
Mike Jenkins	86
John Smith	47
Jen Wu	92

Use of Text

Cisc 124 Grades	
Nick Graham	63
Mike Jenkins	86
John Smith	47
Jen Wu	92

- Colour now used to group name and grade together
- Alphabetical ordering aids looking up names

Presenting Text

- Case
  - Use conventional upper/lower
- Spacing
  - With **`constant-width fonts`**, use ragged-right
  - With proportional-width fonts, use justified
- Spacing between lines
  - letter height : total space between lines
    - 1:2 faster than 1:4
    - 1:2.7 faster than 1:1.3
    - …implies 1:2 to 1:2.7 optimal

Presenting Text

- Line length
  - 40 - 60 characters optimal
  - Inter-line spacing : Line length = 1:30
- Font
  - Just because you have 1,000 fonts doesn't mean you have to use them all
  - Sans-serif easier to read on displays
  - Serif more traditional for longer texts

Graphics

- A picture is worth 1,000 words
- Represent
  - Real world images
  - Complex real-world systems
  - Numerical data
    - Graphical data to represent overall trend; textual for real analysis
  - Direct manipulation objects

ELEC 372

Engineering Software Structures

Week 11: Usability Engineering
Lecture 3

Design of Interactive Systems

- Design is a creative process
- Goal is to design a system that is
  - Appropriate to users in their context
    - *cf* User Needs Analysis
  - Allows users to perform their tasks
    - *cf* Task Model
- Balance usability properties
  - Ease of learning, ease of use

Conceptual/Mental Models

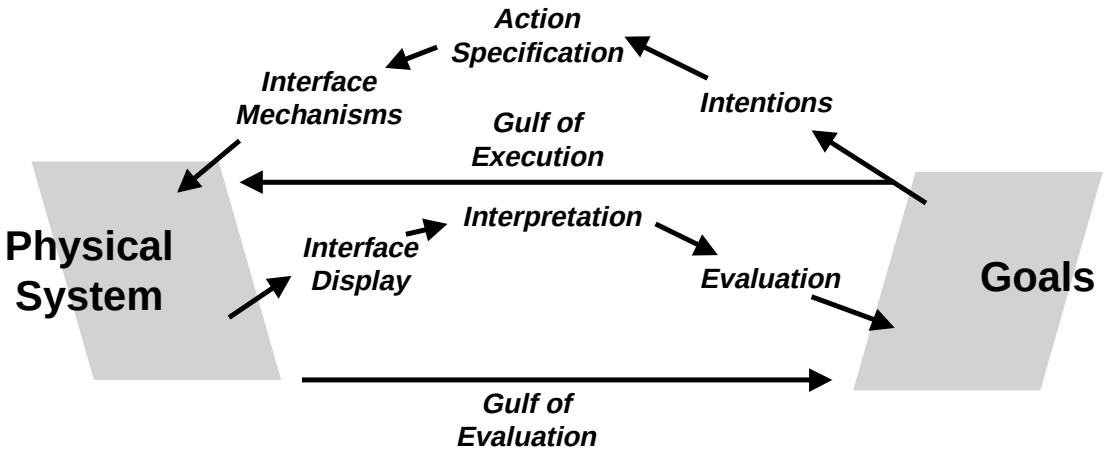
[Norman, Design of Everyday Things]

- Users interacting with systems wish to achieve some goal
- Must find ways of achieving this goal via the mechanisms of the system
- Must understand system in order to
  - Go from goal to sequence of actions to achieve goal
  - Determine whether goal has been achieved by examining system state

Norman's Gulfs of Execution/Evaluation

- Example: pouring water into a bathtub
- Goal: have water flow at desired rate, temperature
- Goal expressed in terms of *psychological variables*
  - Temperature of water
  - Flow rate of water
- Mechanisms available are *physical variables*
  - Hot tap
  - Cold tap
- User must work out how to adjust hot and cold tap to achieve desired values of psychological variables, and must touch/see/hear water to determine whether these values achieved

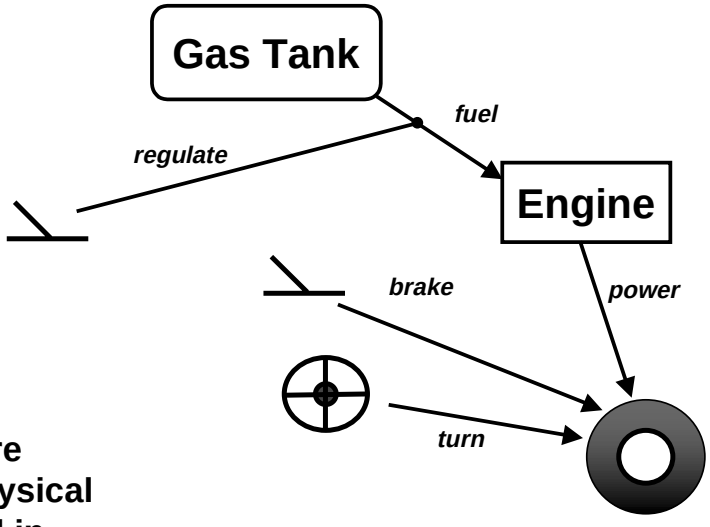
Norman's Gulfs of Execution/Evaluation



Conceptual Models

- In order to accomplish goals with a system, the user needs to understand:
  - What actions on the system are possible
  - What the effects of a particular action will be
  - How to view the state of the system to evaluate the results of actions
- Therefore users need a model of how the systems work
  - Conceptual models allow full, complete, correct understanding of system
  - Used by engineers, teachers

Conceptual Model of Car



Question: What are psychological, physical variables involved in driving a car?

Conceptual Model of Car

- Helps answer questions like:
  - What are effects of controls (accelerator, brake, steering wheel) on speed, direction of car?
  - What is effect of depressing accelerator and brake at same time?
  - What happens if car runs out of gas?
- Explains what we need to know to *operate* a car, not to build one
  - Therefore not an implementation model
    - "The' steering wheel' turns the 'steering column', which acts on a gear in the 'steering rack' (for newer cars with rack and pinion steering) which connects to a steering arm on each side of the car to each 'wheel hub'. Older cars (like my Alfa and the RX-7) have a 'rotating ball' assembly that moves a steering arm back and forth, which pulls one wheel while pushing the other side out" [Dave Phillips, Car Expert]

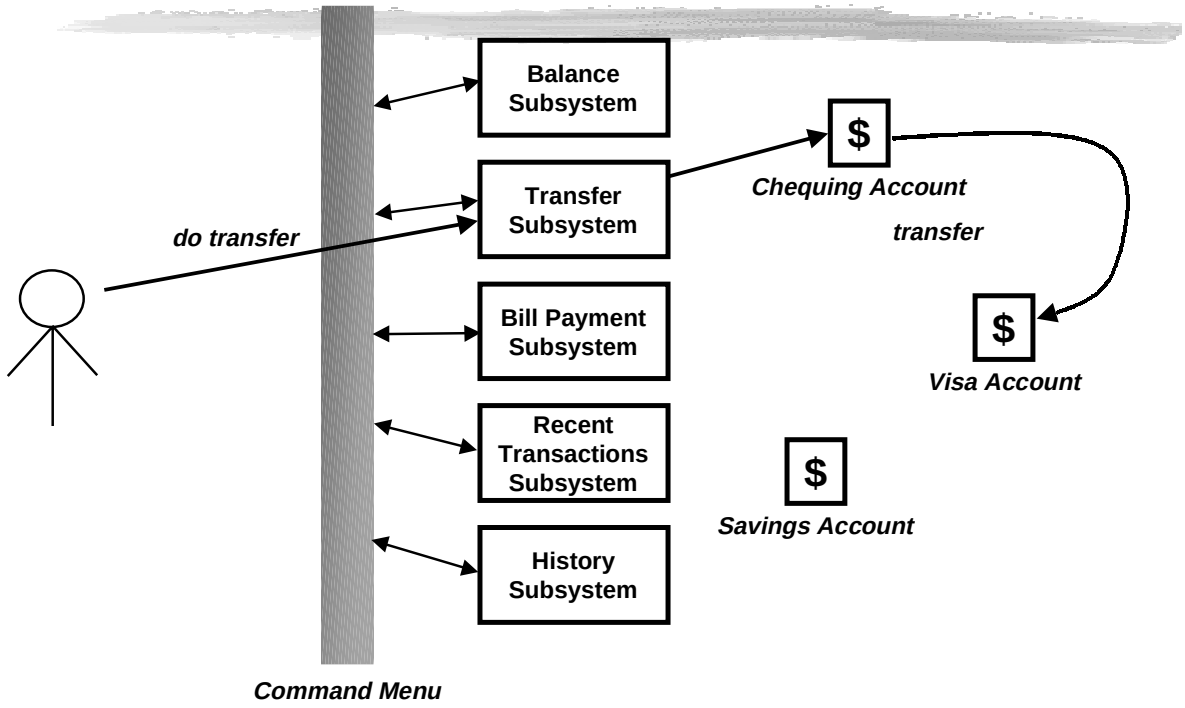
Mental Models

- In learning about a system, users are trying to develop an understanding of its conceptual model
  - Through training, books, experimenting with system
  - User's model called *mental model*
- If conceptual model not visible to users, they will have to guess
- Incorrect mental models lead to difficulties in establishing task-action mapping

Mental Models

- How user understands user interface
- Predictive, explanatory
- Often incorrect, incomplete
- Parsimonious -- kept small enough to understand what features the user needs/uses
- Superstition -- if something doesn't work, easier to develop superstition than to extend mental model
- People are loathe to extend their mental models

Conceptual Model of Banking System



Developing Mental Model

- Navigation structure very clear
  - Main menu makes this part of conceptual model visible
- Familiar concept of *account* revealed in conceptual model
- Need to experiment with system to discover that visa is an account not a bill
  - Therefore, pay visa via transfer, not bill payment
- Distinction between "history" and "recent transactions" unclear
  - From experimentation, history gives more transactions
  - Recent transactions may be updated faster after a transaction is performed (but this may be just superstition)

Techniques for Stimulating Creative Design

- Metaphor
- Brainstorming

Metaphors

- Something familiar to user from real world
- Helps to structure user interface
- Helps user understand structure, predict behaviour of user interface
  - ∴ Helps in developing mental model

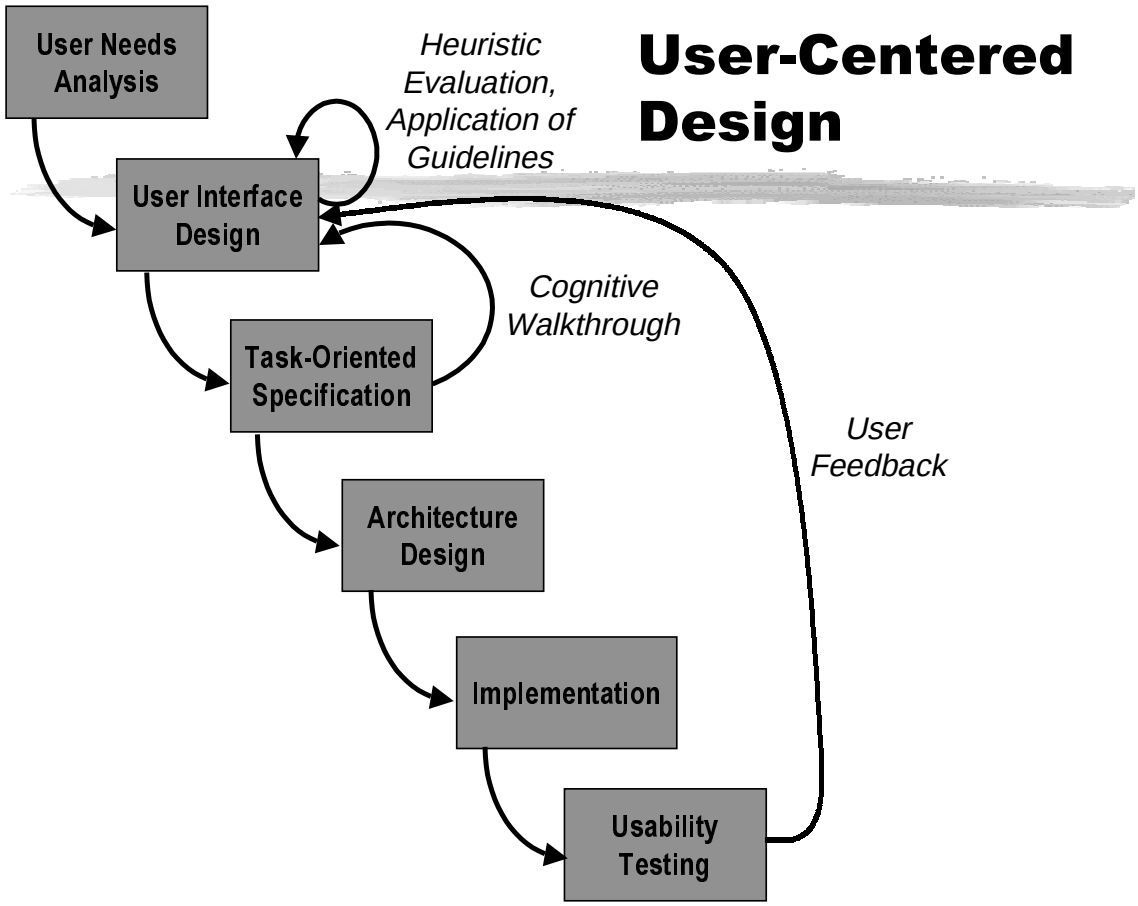
ELEC 372

Engineering Software Structures

Week 11: Usability Engineering
Lecture 4

Evaluating User Interfaces

- Important to make evaluation early, continuous
- Techniques:
  - Heuristic evaluation
  - Guidelines
  - Cognitive walkthrough
  - Usability testing (user observation)



Heuristic Evaluation

- User interface specialists study user interface design in depth
- Looking for known problems from their experience

Guidelines

- Guidelines encode experience
- In effect, allow non-specialists to perform their own heuristic evaluation
  - Guidelines help structure system evaluation by telling designers what problems to look for

Cognitive Walkthrough

- Walk through interface in context of user trying to perform a task
  - Identify core tasks
  - Show how they would be accomplished by walking through interaction steps with user interface
- Shows difficulties with task-action mapping
  - E.g. task hard to perform, task not supported at all
- Helps identify areas where operations should be optimized

Approaches to Cognitive Walkthrough

- Informal
  - Narrative scenario
    - write out steps followed by user in narrative prose
  - Low-fidelity prototype
    - cutouts of interface elements that can be manipulated by hand
  - Mockups
    - Simulations in Macromedia Director, Shockwave etc
- Formal
  - Precise scenario
    - UML sequence diagram
  - Task-oriented specification
    - Refine ConcurTaskTree to action level
- Formal approaches are great, but significant information can be gained from cheap, informal approaches

Usability Testing

- Test usability of system by observing users
- Users given specific task to accomplish
- Observe to see what problems arise
- Can be performed with low fidelity prototype or with real system
  - Trade off quality of feedback vs how early it can be obtained

Comparison of Four Approaches

- UWE Project, HP [Jeffries et al., 1991]
- Visual interface to Unix under X/Motif
- Applied four methods separately to compare results
- Error types
  - Consistency (25% of problems)
  - Recurring -- one time or recurring?
  - General -- affecting several parts of UI or specific to single part?

Heuristic Evaluation	*Usability Testing*	*Guidelines*	*Cognitive Walkthrough*
105 core errors identified	31 core errors identified	35 core errors identified	35 core errors identified
– Identifies more problems – Identifies more serious problems – Low cost – Doesn't require implementation	– Identifies serious, recurring problems – Avoids low priority problems	– Identifies recurring, general problems – Can be used by software developers – Does not require implementation	– Helps define user's goals, assumptions – Can be used by software developers – Does not require implementation
– Requires UI expertise – Requires several evaluators	– Requires UI expertise – High cost – Misses consistency problems – Requires implementation	– Misses some severe problems	– Needs task model – Tedious – Misses general, recurring problems

Kinds of Users for Testing
[Booth]

- Friendly users
  - Some technical knowledge
  - Can make suggestions that naïve users perhaps could not
- Hostile users
  - People with no investment in system
  - May be useful at finding ways of breaking system
- Simulating users
  - If progress of naïve users recorded, can be later replicated by designers
  - Form of informed cognitive walkthrough
  - Saves expense of live users for incremental modifications of system

Some ways of describing a user interface

- Screen mockups
- Scenarios
- Task-oriented specification

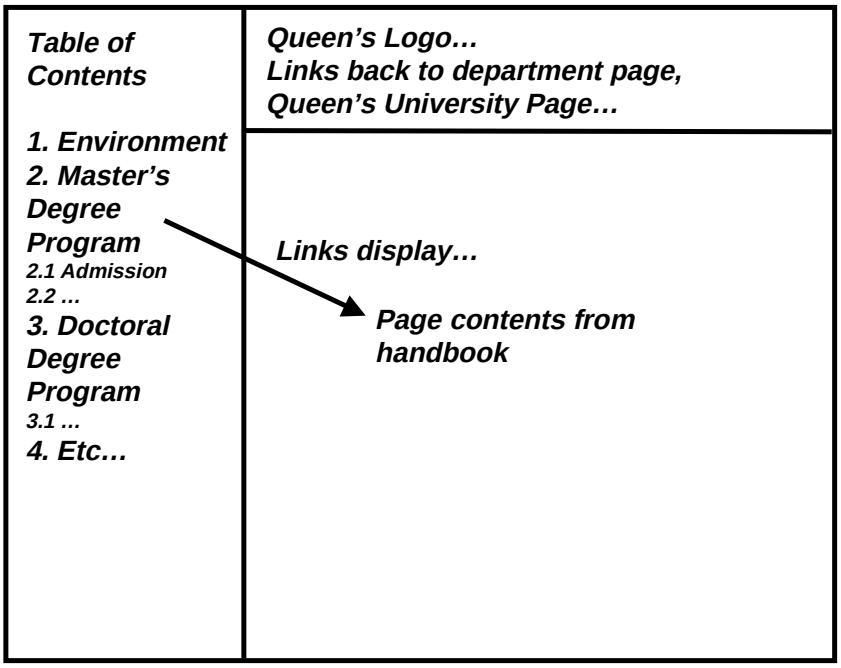
E.g.: User interface for finding out about courses offered

- Tasks supported include:
  - Find out what courses offered
  - Find out who is teaching them
  - Find any scheduling conflicts
  - Find out about content of courses
  - Find out workload of course selection

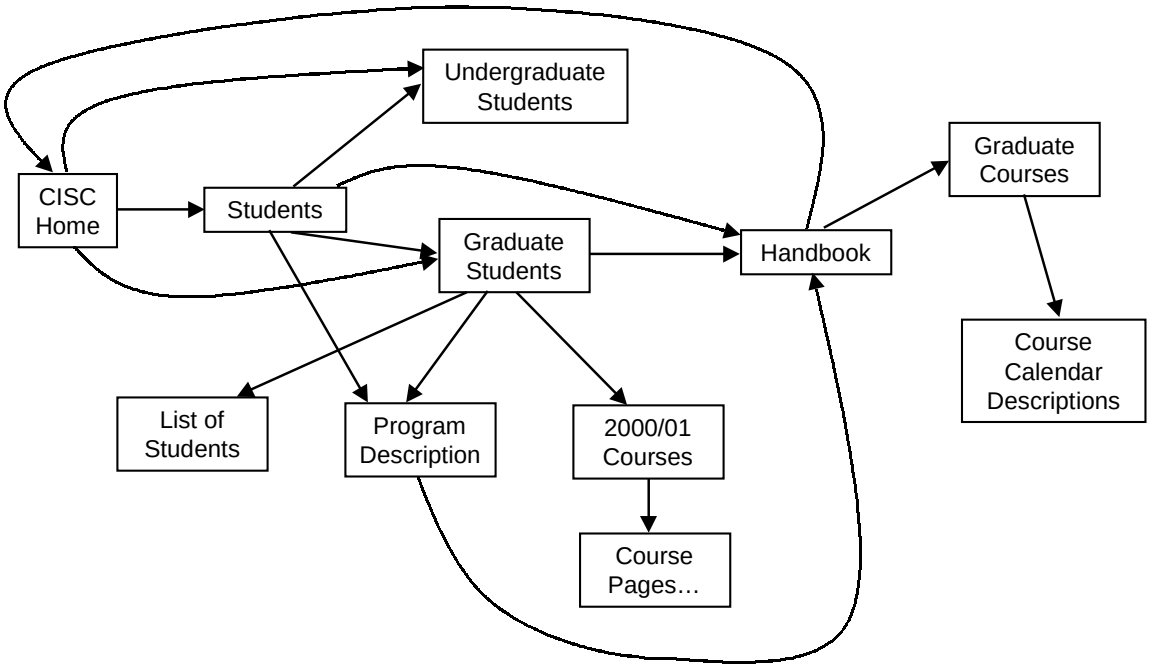
Mockups

- Low fidelity
  - Pen and paper drawings
  - Rough sketches with drawing package
- High fidelity
  - Professional drawings with details included
  - Animations: motion GIFS, Macromedia Director, Flash
- Prototypes
  - Visual Basic, Graphical mockup in Visual Builder

E.g., Low-Fi Mockup of Web Page



E.g., navigation structure of screen-based application (much abbreviated)



Scenarios

- Tell a story to describe
  - how system used
  - interaction design of system
- Many different kinds
  - Narrative scenario
  - Storyboard
  - Precise scenario (e.g. UML)
- Help in design, explanation, evaluation

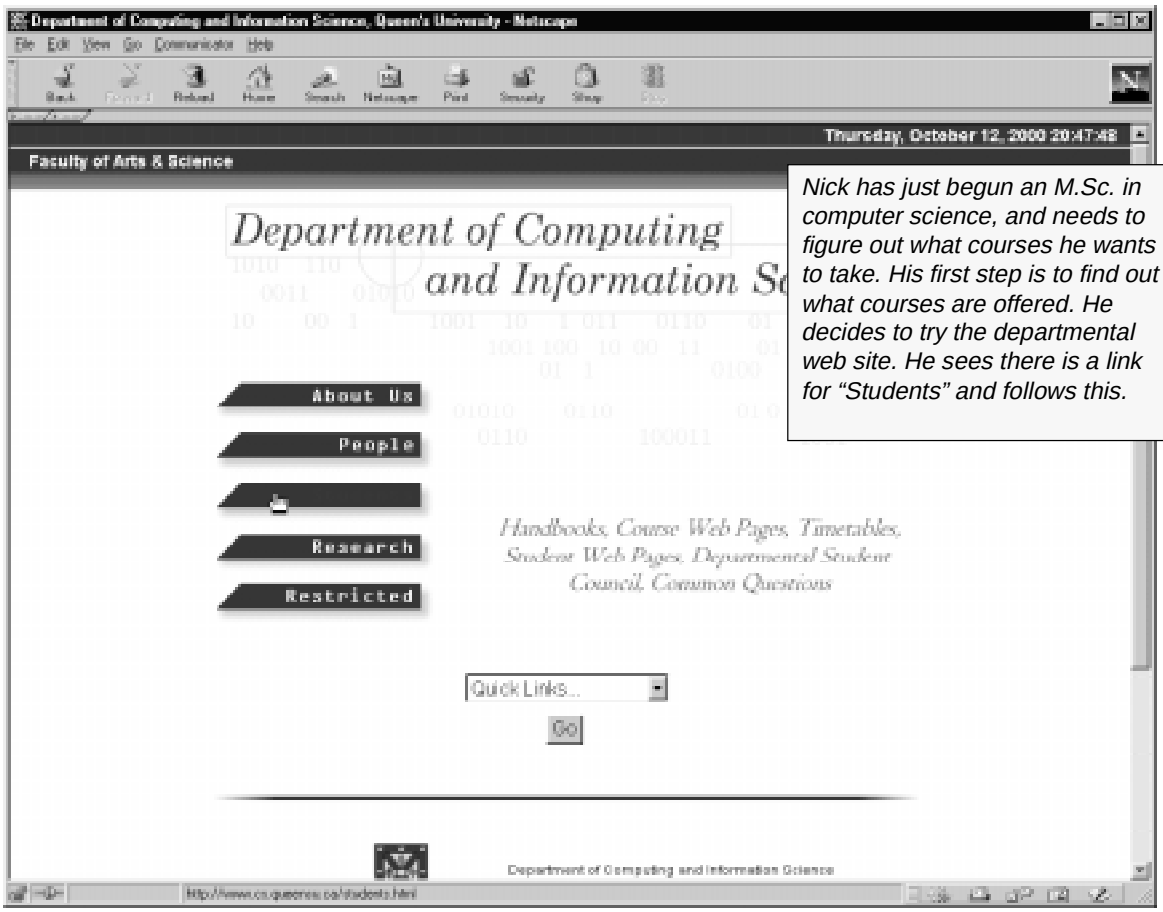
E.g., Narrative Scenario

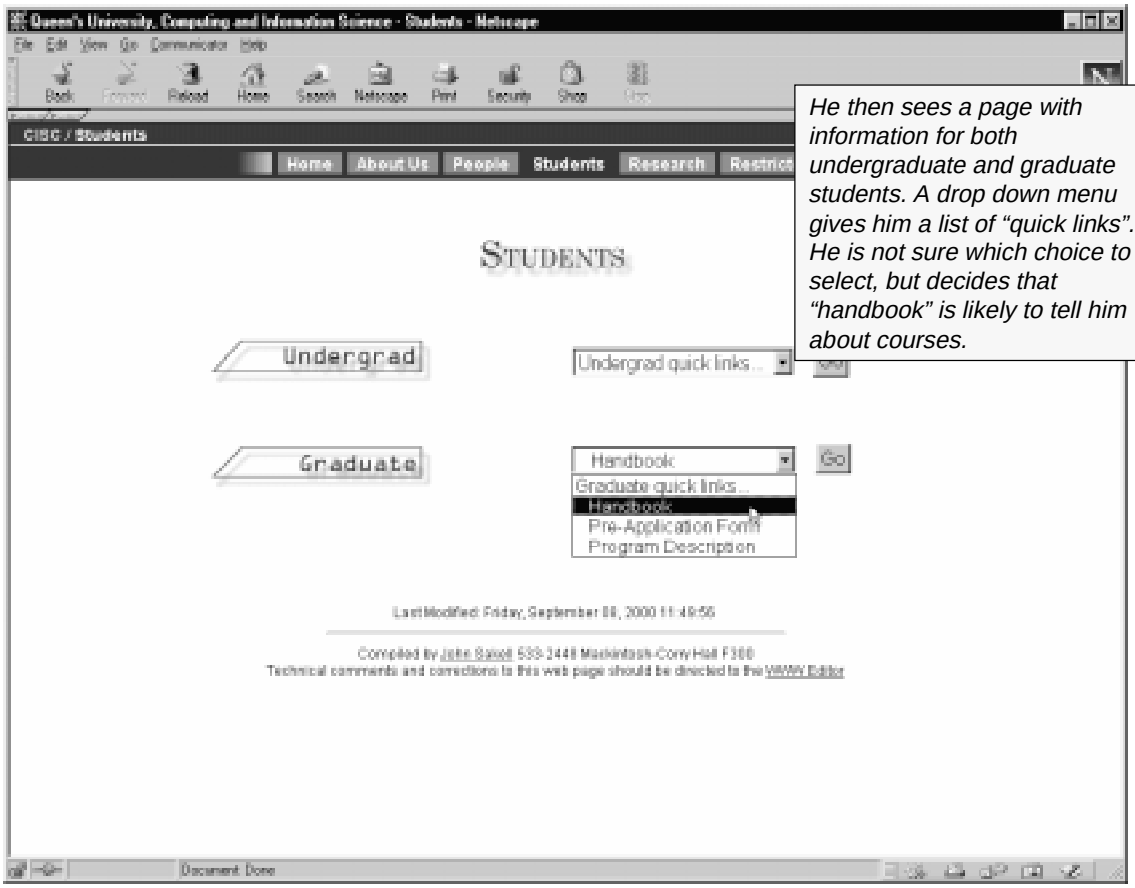
Nick has just begun an M.Sc. in computer science, and needs to figure out what courses he wants to take. His first step is to find out what courses are offered. He decides to try the departmental web site. He sees there is a link for "Students" and follows this. He then sees a page with information for both undergraduate and graduate students. A drop down menu gives him a list of "quick links". He is not sure which choice to select, but decides that "handbook" is likely to tell him about courses. At the handbook page, he finds a long table of contents. In this, he finds an entry for "Graduate Courses". He clicks this, and in the window's main pane, finds a list of all graduate courses. These courses are organized by topic area. He is interested by the idea of theory courses, so clicks on a number of them, and looks at their content. He decides that CISC 867, Algorithm Design and Analysis looks very interesting. However, he is a little worried that the course might be too mathematical, so sends an email to a friend who took the course last year.

Etc……

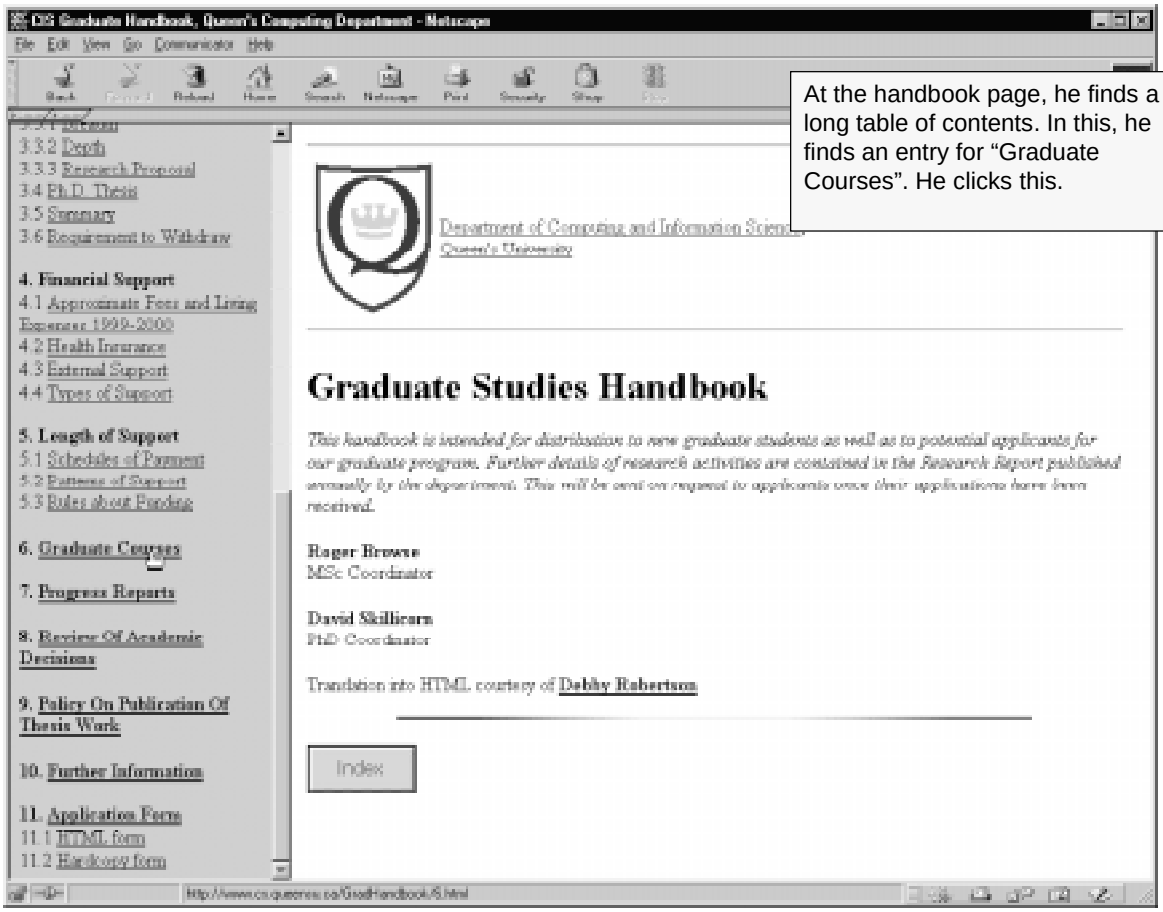
E.g., storyboard

- Same idea as narrative scenario, but illustrated with pictures
- (In this example, I'm using screen snapshots of the real system)

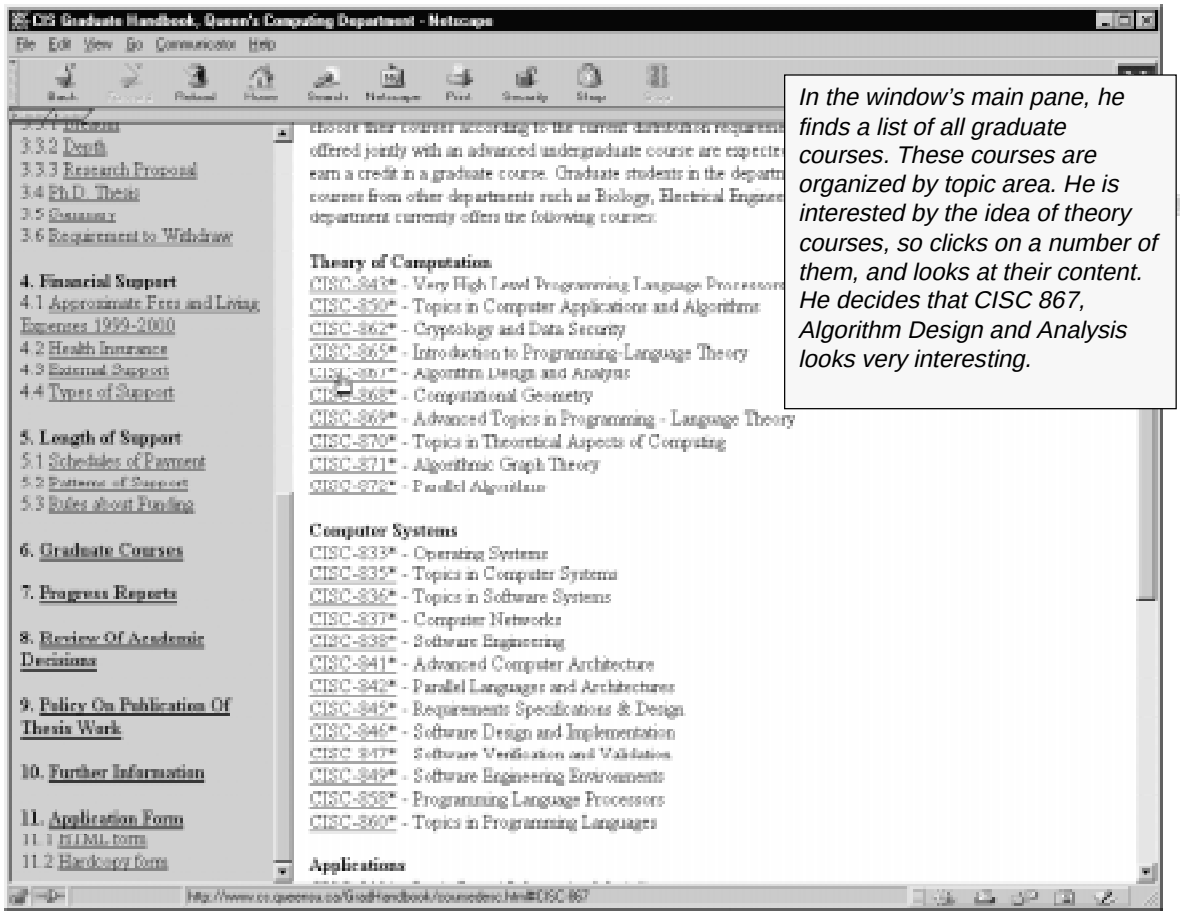




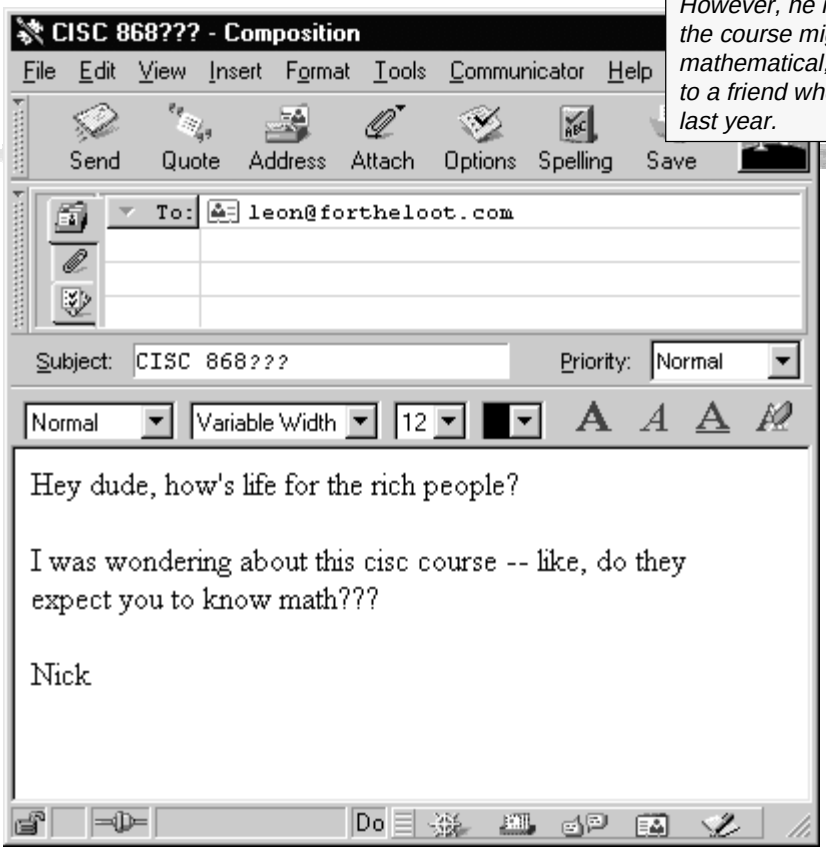
He then sees a page with information for both undergraduate and graduate students. A drop down menu gives him a list of "quick links". He is not sure which choice to select, but decides that "handbook" is likely to tell him about courses.



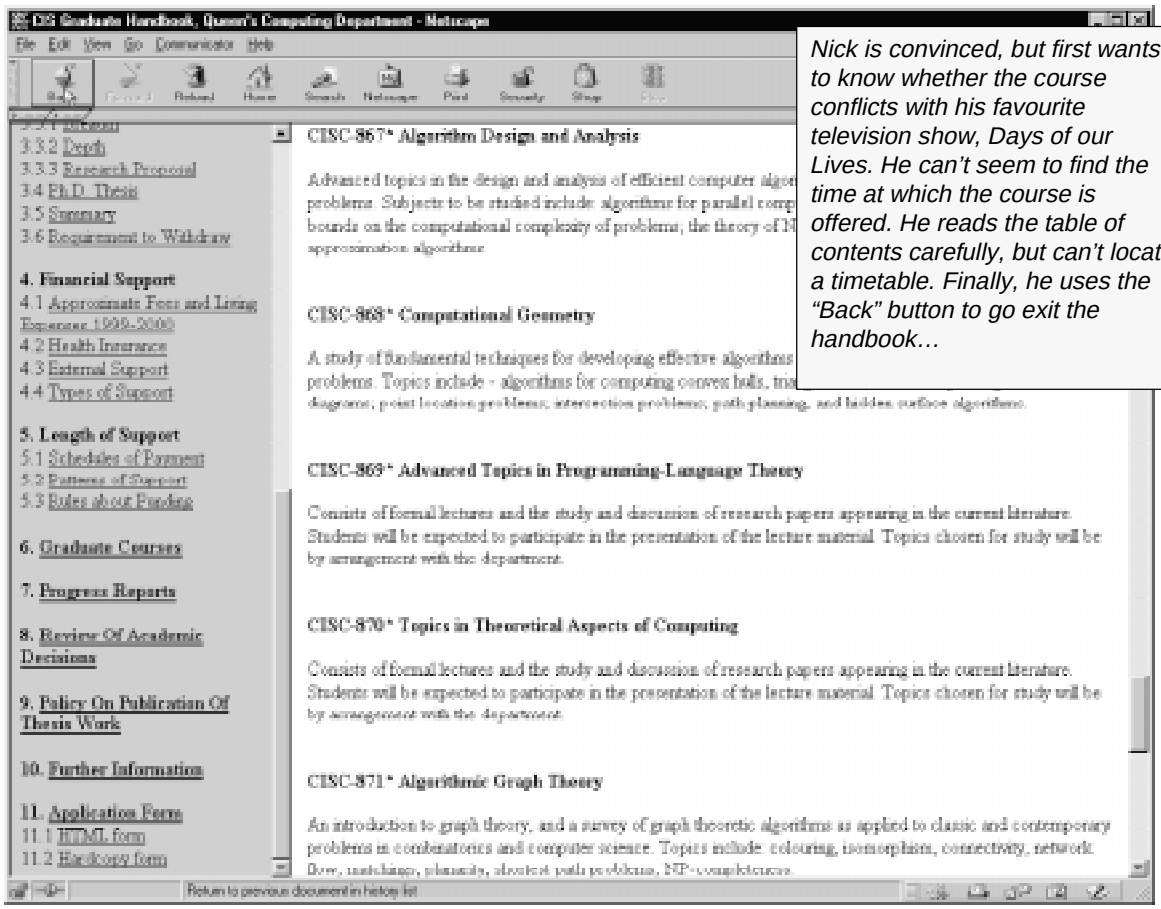
At the handbook page, he finds a long table of contents. In this, he finds an entry for "Graduate Courses". He clicks this.



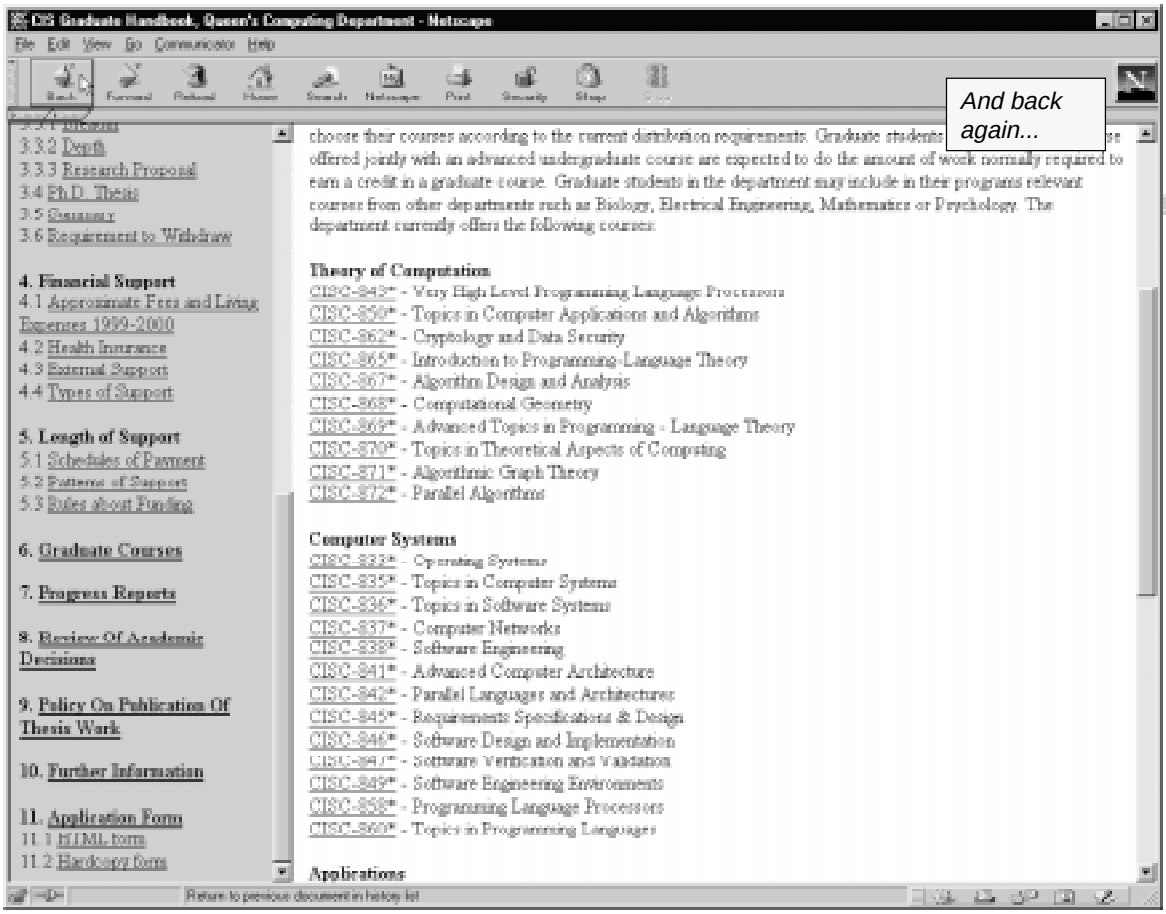
In the window's main pane, he finds a list of all graduate courses. These courses are organized by topic area. He is interested by the idea of theory courses, so clicks on a number of them, and looks at their content. He decides that CISC 867, Algorithm Design and Analysis looks very interesting.



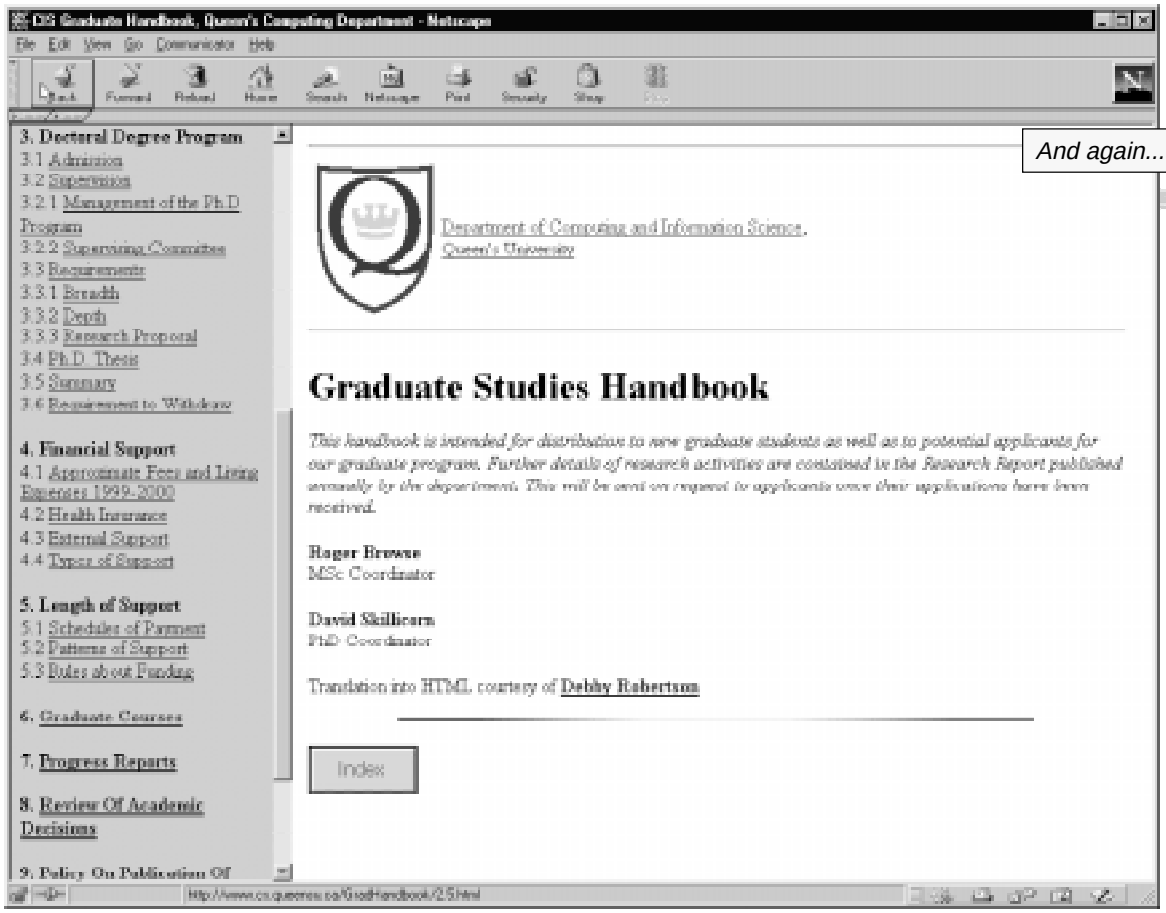
However, he is a little worried that the course might be too mathematical, so sends an email to a friend who took the course last year.



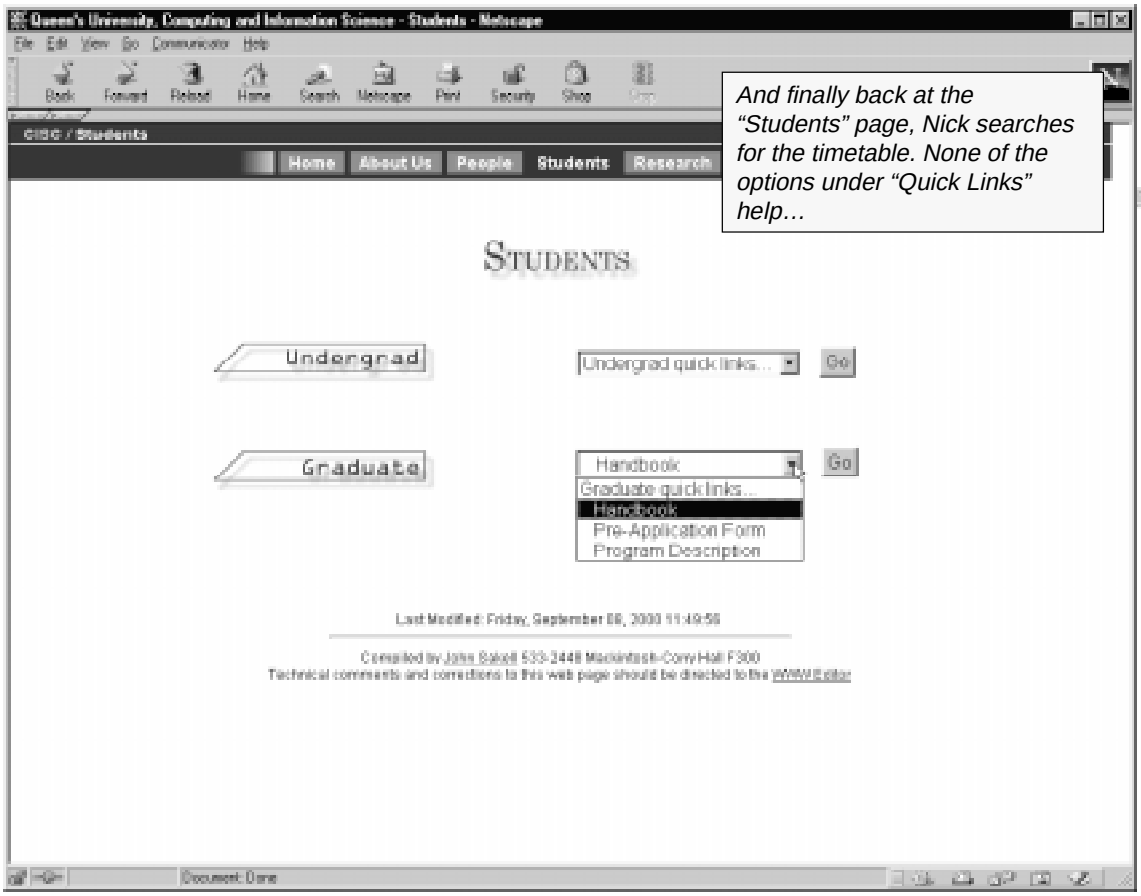
Nick is convinced, but first wants to know whether the course conflicts with his favourite television show, Days of our Lives. He can't seem to find the time at which the course is offered. He reads the table of contents carefully, but can't locate a timetable. Finally, he uses the "Back" button to go exit the handbook…



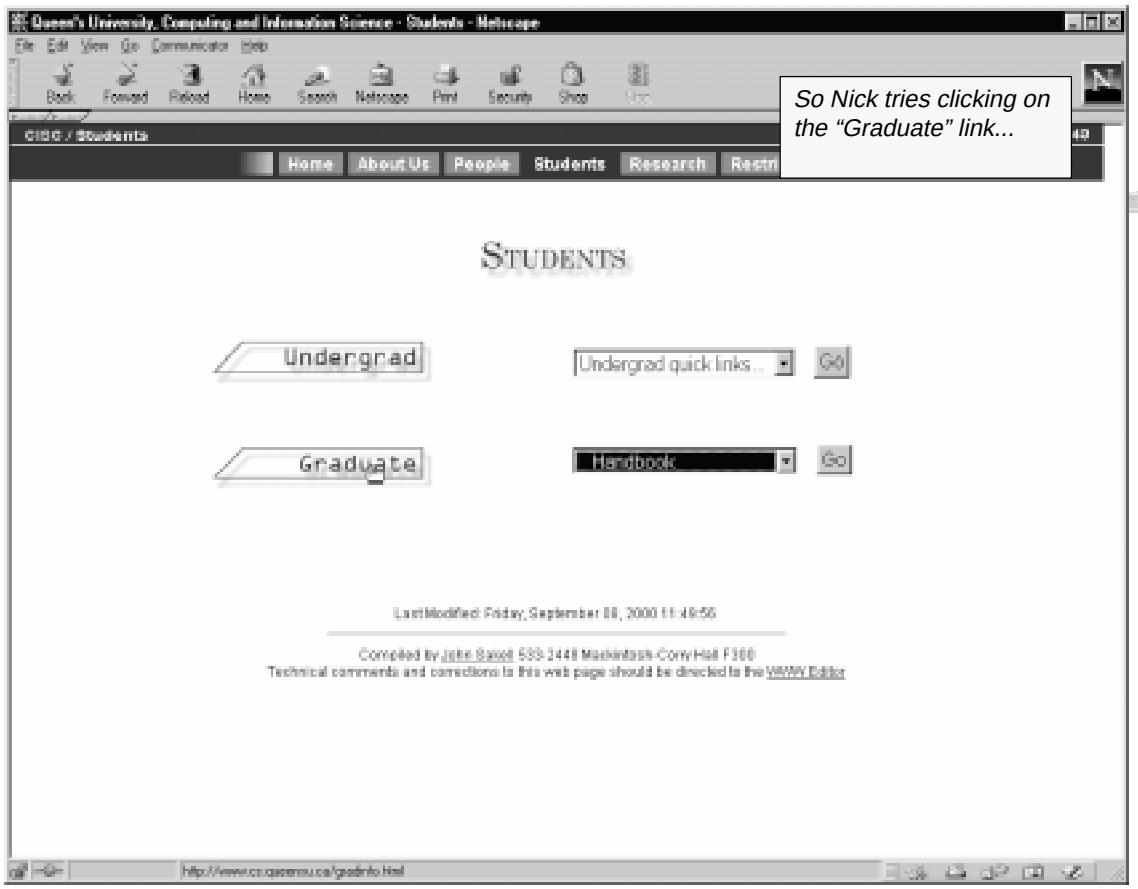
And back again...

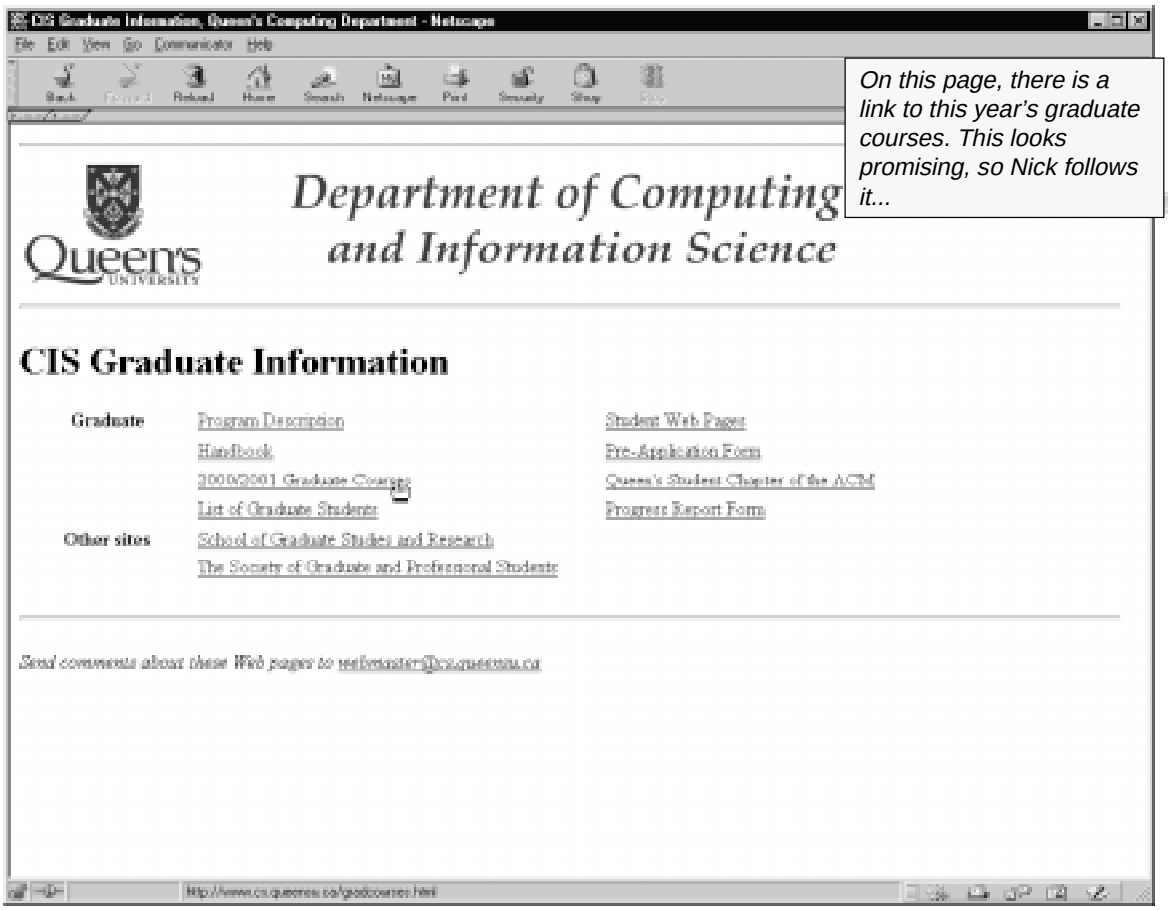


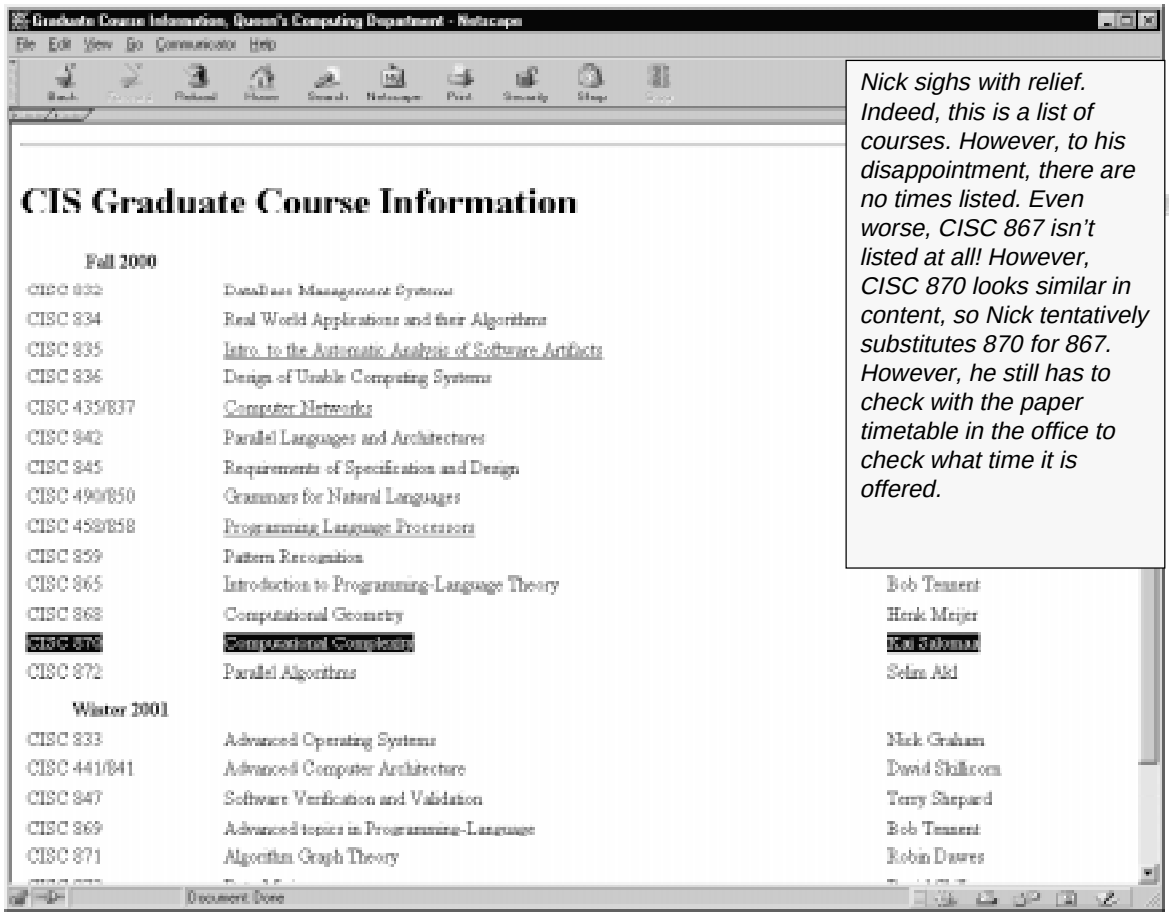
And again...



And finally back at the "Students" page, Nick searches for the timetable. None of the options under "Quick Links" help…


Queen's University, Computing and Information Science - Students - Netscape
CISC / Students
Home
About Us
People
Students
Research
STUDENTS
Undergrad
Undergrad quick links...
Go
Graduate
Handbook
Go
So Nick tries clicking on the "Graduate" link...


CIS Graduate Information, Queen's Computing Department - Netscape
Department of Computing and Information Science
CIS Graduate Information
Graduate
Program Description
Handbook
2000/2001 Graduate Courses
List of Graduate Students
Other sites
School of Graduate Studies and Research
The Society of Graduate and Professional Students
Student Web Pages
Pre-Application Form
Queen's Student Chapter of the ACM
Progress Report Form
Send comments about these Web pages to webmaster@cs.queensu.ca
On this page, there is a link to this year's graduate courses. This looks promising, so Nick follows it...



Nick sighs with relief. Indeed, this is a list of courses. However, to his disappointment, there are no times listed. Even worse, CISC 867 isn't listed at all! However, CISC 870 looks similar in content, so Nick tentatively substitutes 870 for 867. However, he still has to check with the paper timetable in the office to check what time it is offered.

Task-Oriented Specification

- Forms bridge between point of view of designer and point of view of implementer
- Provides specification of how tasks carried out with particular system
- Allows analysis of
  - Complexity of task-action mapping
  - Task coverage
  - Consistency
- Sometimes called *System Task Model*

User-Centered Design

