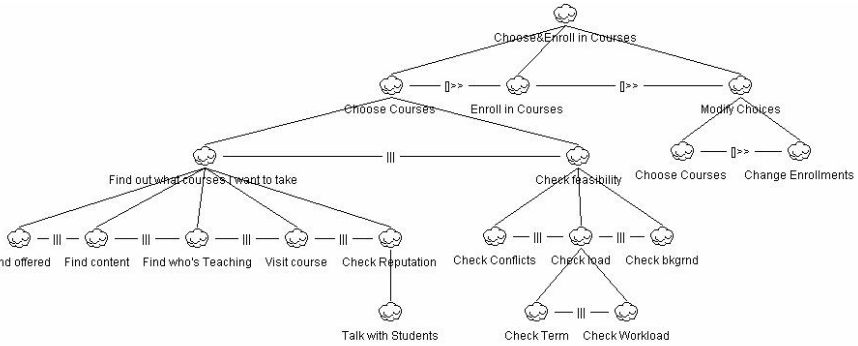
CISC 839
Software Engineering of Usable Computing Systems

T.C. Nicholas Graham
http://stl.cs.queensu.ca/~graham/cisc836

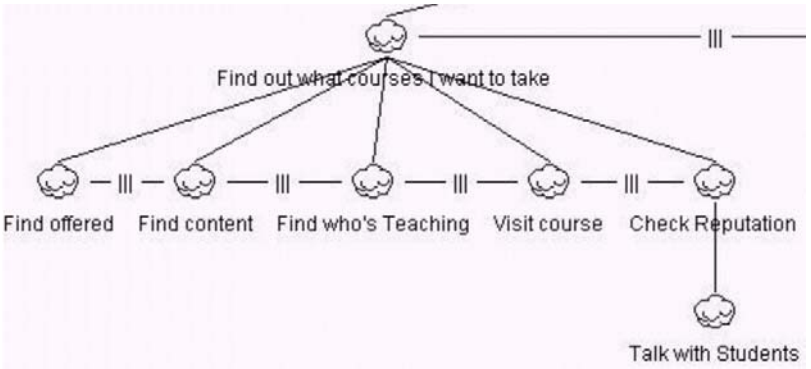
Some ways of describing a user interface

- Screen mockups
- Scenarios
- Task-oriented specification

Hierarchical Task Model for Course Selection+Enrollment

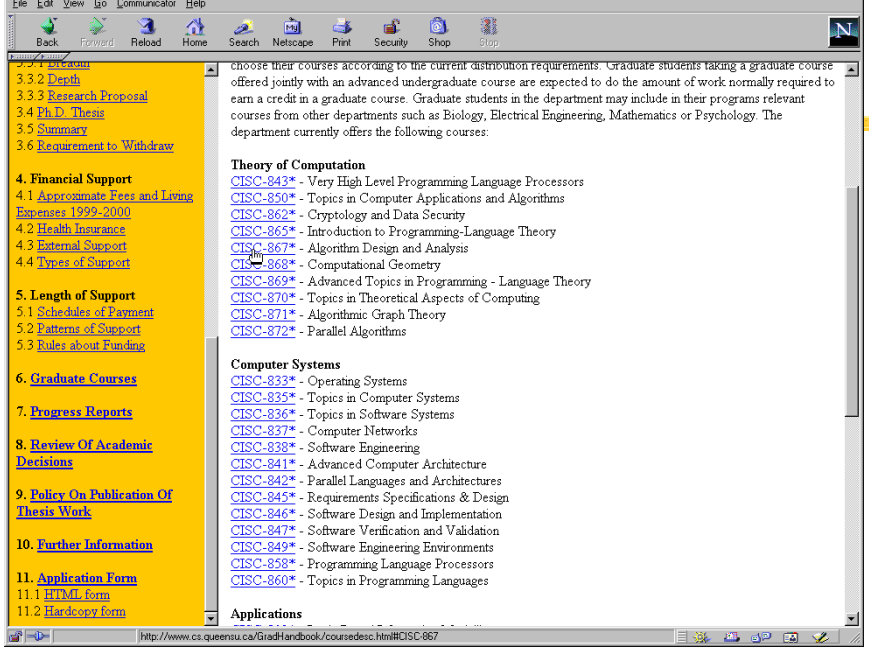


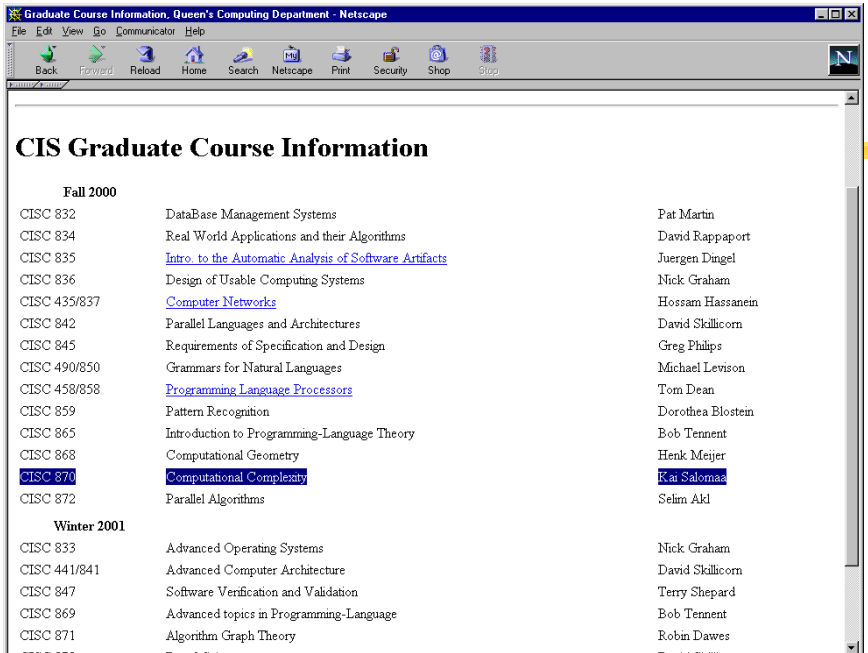
Example: existing interface for system to "find out what courses I want to take": `http://www.cs.queensu.ca`



Current Interface

- Web pages give
  - List of courses
  - List of courses offered this year
  - Some courses have web pages with details
- Course schedule is on paper

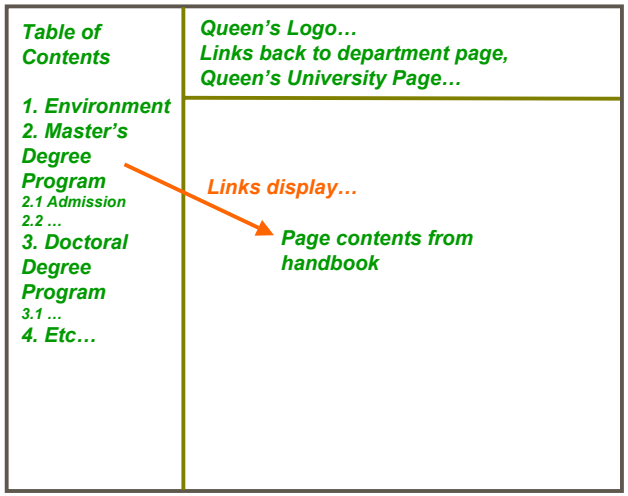




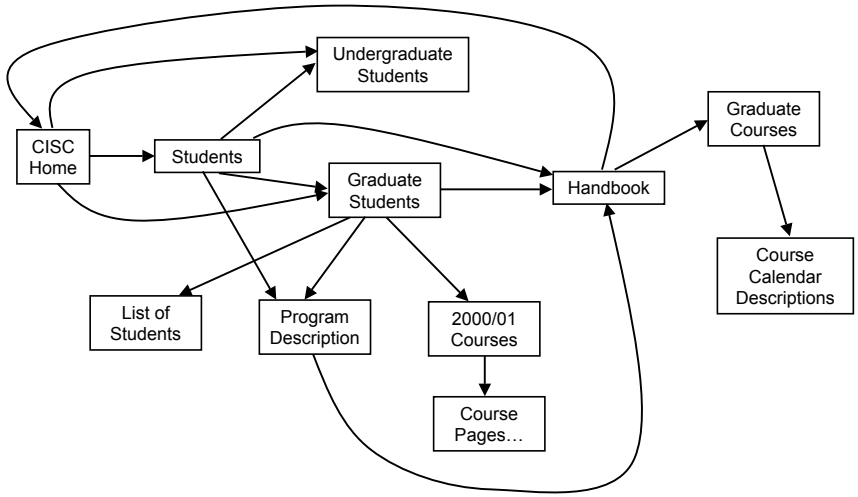
Mockups

- Low fidelity
  - Pen and paper drawings
  - Rough sketches with drawing package
- High fidelity
  - Professional drawings with details included
  - Animations: motion GIFS, Macromind Director
- 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

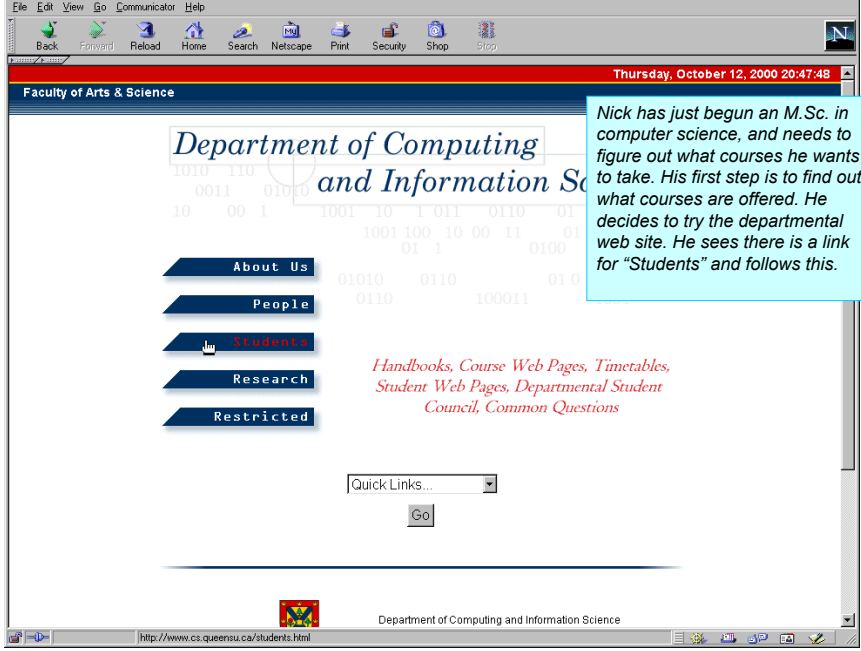
E.g., Narrative Scenario

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.

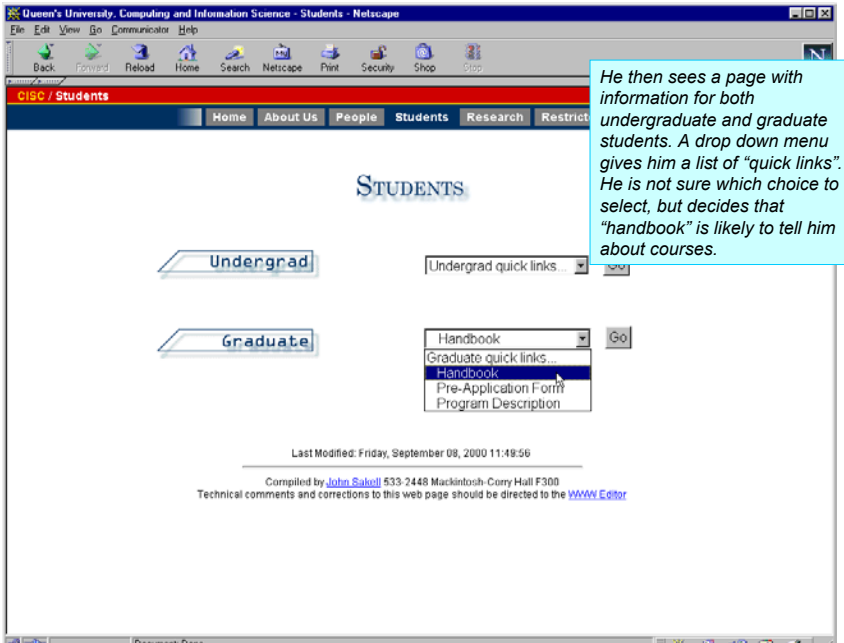
Etc.....

E.g., storyboard

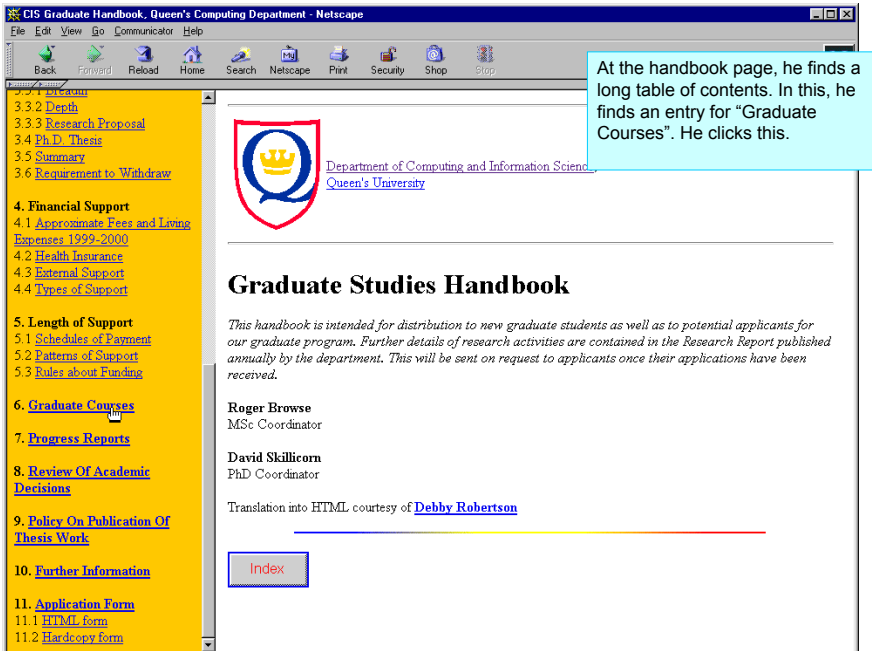
- Same idea as narrative scenario, but illustrated with pictures
- (In this example, I'm using screen snapshots of the real system)



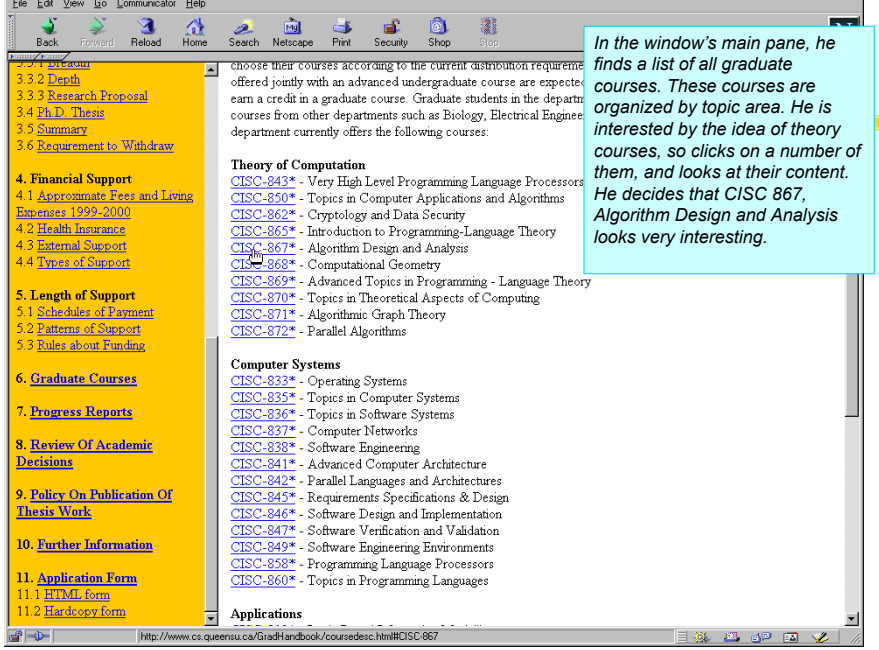
Nick has just begun an M.Sc. in computer science, and needs to figure out what courses he wants to take. His first step is to find out what courses are offered. He decides to try the departmental web site. He sees there is a link for "Students" and follows this.



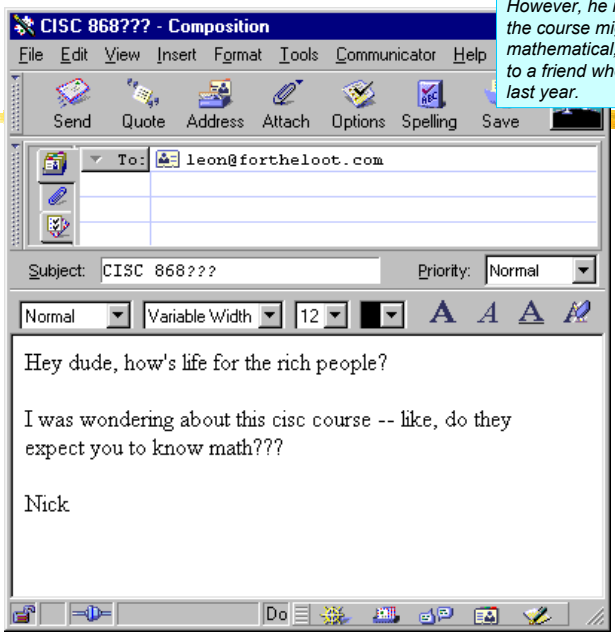
He then sees a page with information for both undergraduate and graduate students. A drop down menu gives him a list of "quick links". He is not sure which choice to select, but decides that "handbook" is likely to tell him about courses.



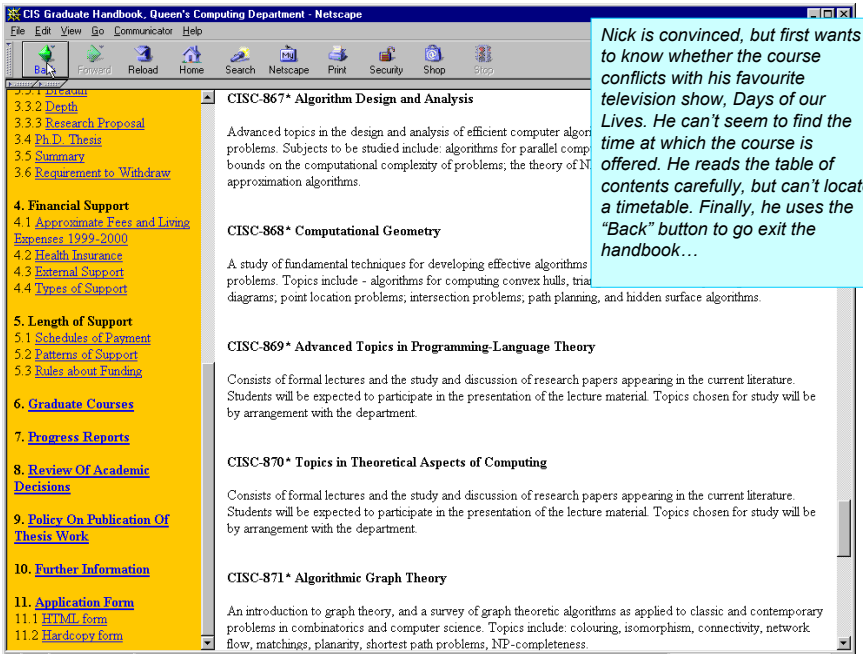
At the handbook page, he finds a long table of contents. In this, he finds an entry for "Graduate Courses". He clicks this.



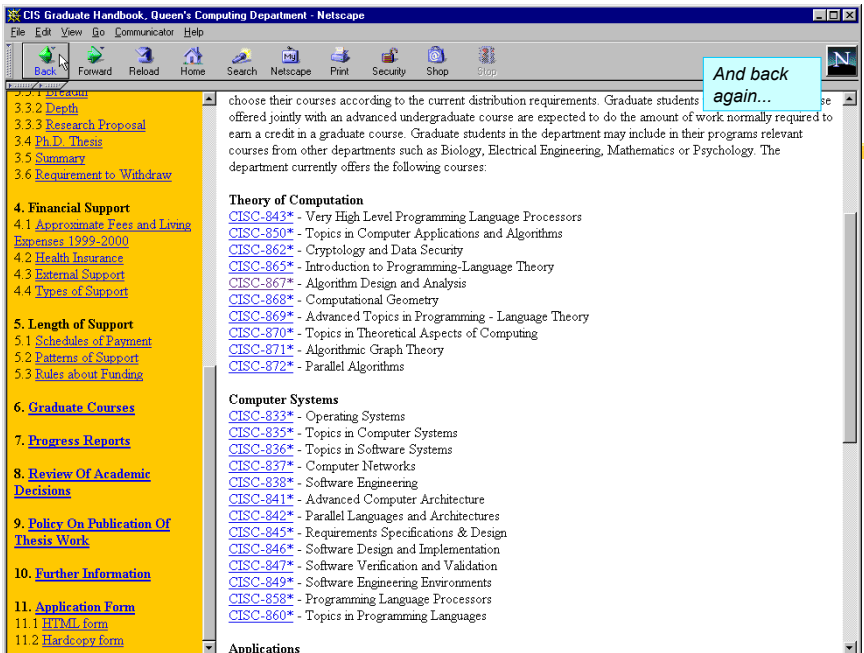
In the window's main pane, he finds a list of all graduate courses. These courses are organized by topic area. He is interested by the idea of theory courses, so clicks on a number of them, and looks at their content. He decides that CISC 867, Algorithm Design and Analysis looks very interesting.



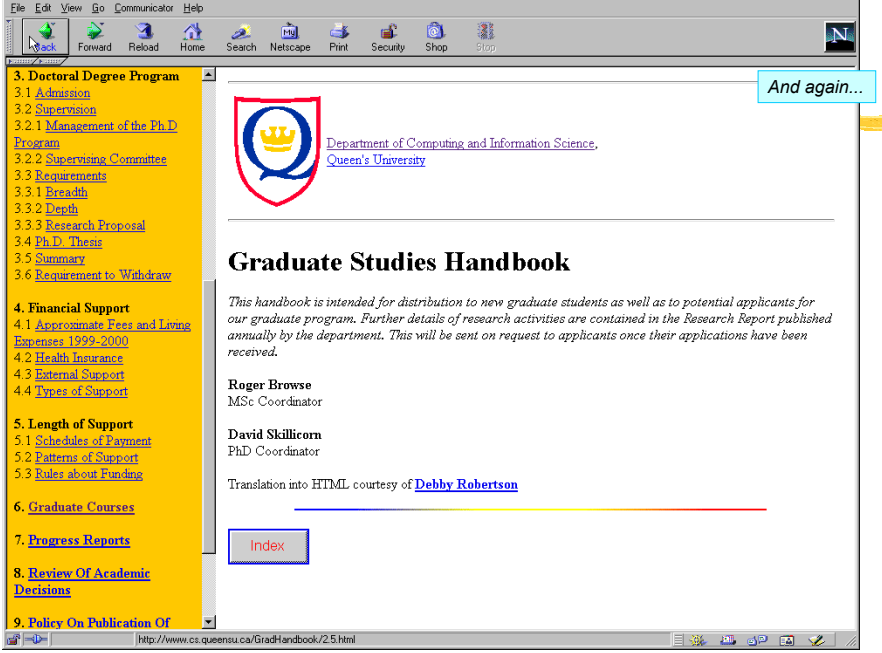
However, he is a little worried that the course might be too mathematical, so sends an email to a friend who took the course last year.

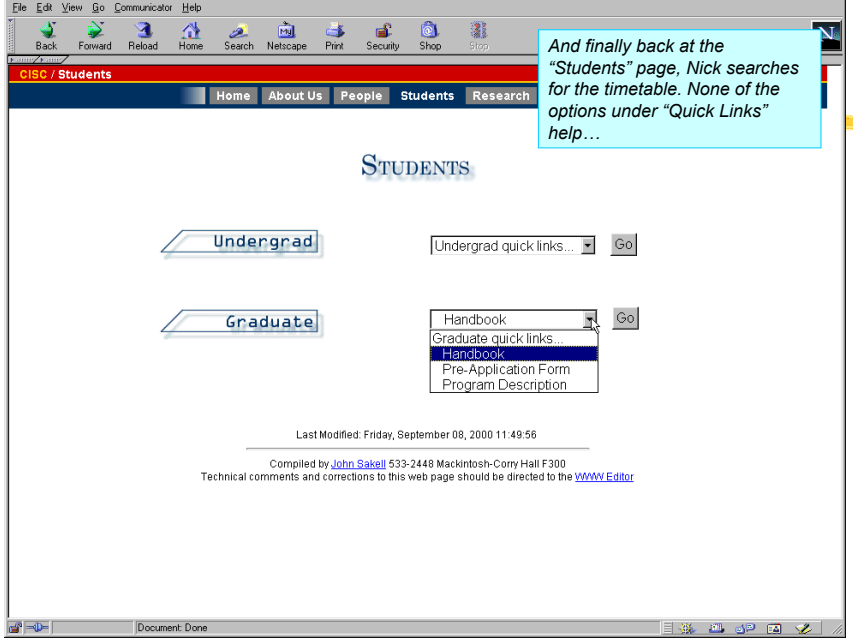


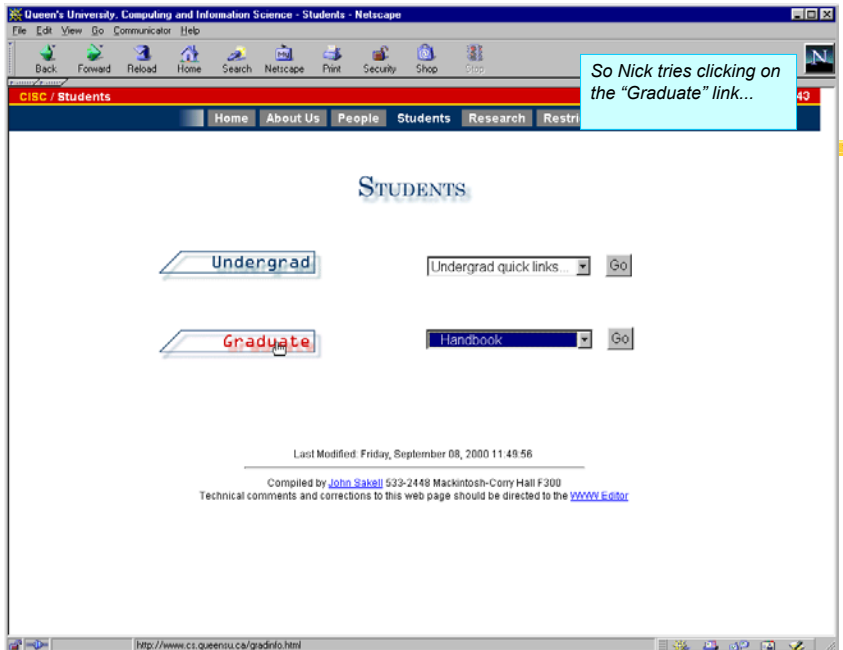
Nick is convinced, but first wants to know whether the course conflicts with his favourite television show, Days of our Lives. He can't seem to find the time at which the course is offered. He reads the table of contents carefully, but can't locate a timetable. Finally, he uses the "Back" button to go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

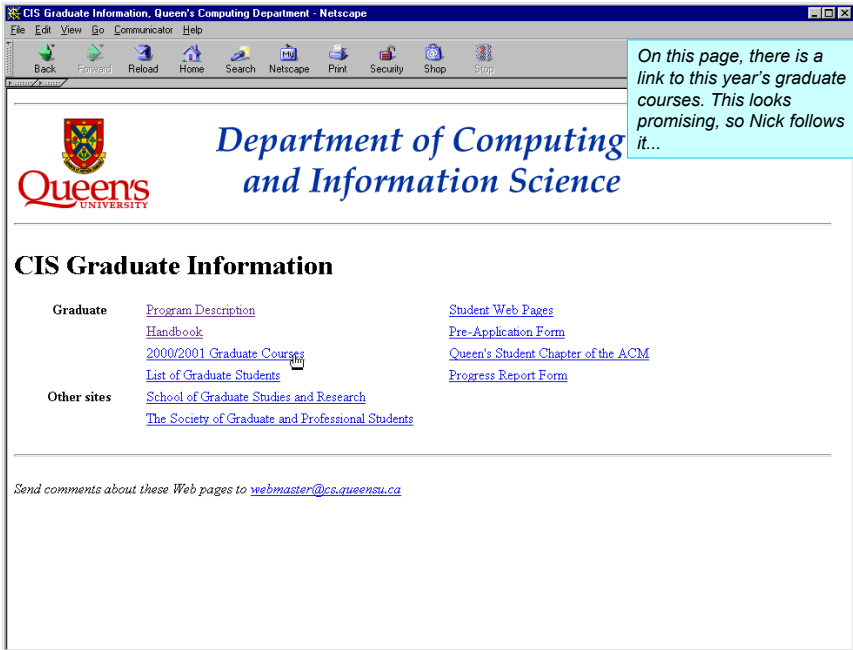


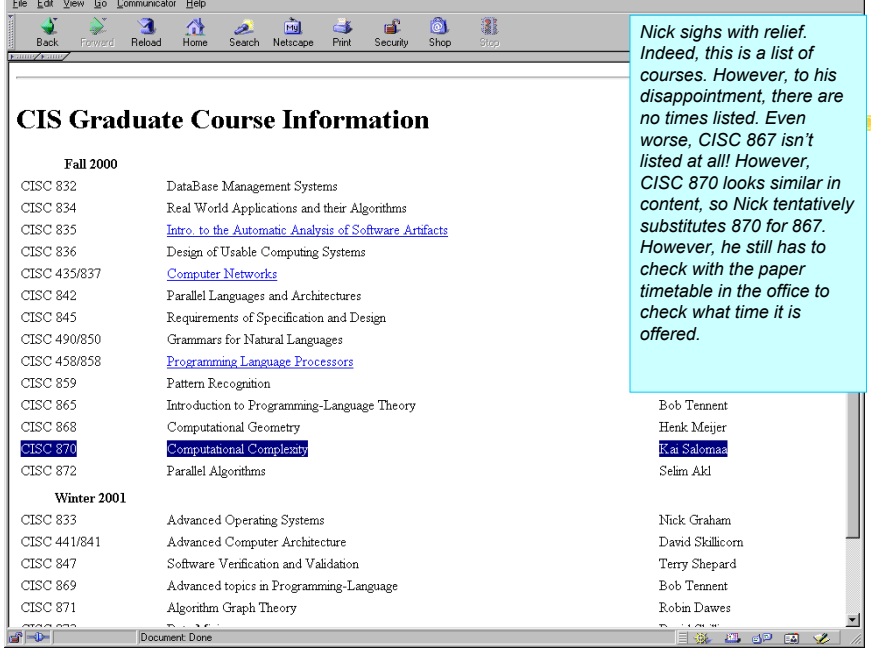
And back again…


File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
3. Doctoral Degree Program
3.1 Admission
3.2 Supervision
3.2.1 Management of the Ph.D Program
3.2.2 Supervising Committee
3.3 Requirements
3.3.1 Breadth
3.3.2 Depth
3.3.3 Research Proposal
3.4 Ph.D. Thesis
3.5 Summary
3.6 Requirement to Withdraw
4. Financial Support
4.1 Approximate Fees and Living Expenses 1999-2000
4.2 Health Insurance
4.3 External Support
4.4 Types of Support
5. Length of Support
5.1 Schedules of Payment
5.2 Patterns of Support
5.3 Rules about Funding
6. Graduate Courses
7. Progress Reports
8. Review Of Academic Decisions
9. Policy On Publication Of
And again...
Department of Computing and Information Science,
Queen's University
Graduate Studies Handbook
This handbook is intended for distribution to new graduate students as well as to potential applicants for our graduate program. Further details of research activities are contained in the Research Report published annually by the department. This will be sent on request to applicants once their applications have been received.
Roger Browse
MSc Coordinator
David Skillicorn
PhD Coordinator
Translation into HTML courtesy of Debby Robertson
Index
http://www.cs.queensu.ca/GradHandbook/2.5.html


File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
And finally back at the "Students" page, Nick searches for the timetable. None of the options under "Quick Links" help…
CISC / Students
Home About Us People Students Research
STUDENTS
Undergrad
Undergrad quick links...
Go
Graduate
Handbook
Graduate quick links...
Handbook
Pre-Application Form
Program Description
Go
Last Modified: Friday, September 08, 2000 11:49:56
Compiled by John Sakell 533-2448 Mackintosh-Corry Hall F300
Technical comments and corrections to this web page should be directed to the WWW Editor
Document: Done


Queen's University, Computing and Information Science - Students - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
So Nick tries clicking on the "Graduate" link...
CISC / Students
Home About Us People Students Research
STUDENTS
Undergrad
Undergrad quick links...
Go
Graduate
Handbook
Go
Last Modified: Friday, September 08, 2000 11:49:56
Compiled by John Sakell 533-2448 Mackintosh-Corry Hall F300
Technical comments and corrections to this web page should be directed to the WWW Editor
http://www.cs.queensu.ca/gradinfo.html

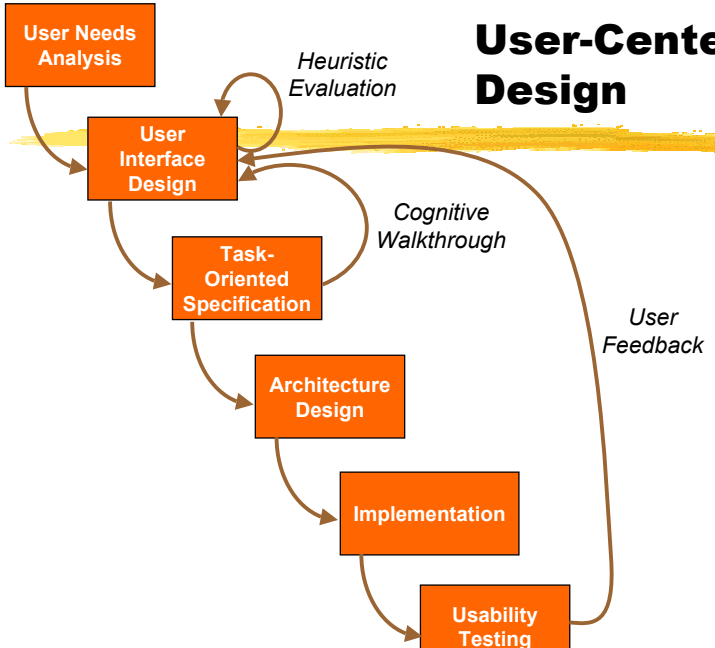

CIS Graduate Information, Queen's Computing Department - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
On this page, there is a link to this year's graduate courses. This looks promising, so Nick follows it...
Queen's University
Department 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



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

Approach

- Start with user task model
- Prune parts of model that will not be handled by system
- Refine model to show how actions carried out with system
- Simply refinement of tasks to include system information
- Represents task-action mapping
  - Cf Norman's gulf of execution

New Task Types

-- Really represent *actions*, not tasks

- User Task
  - Actions carried out completely by the user
  - E.g., thinking, writing on paper

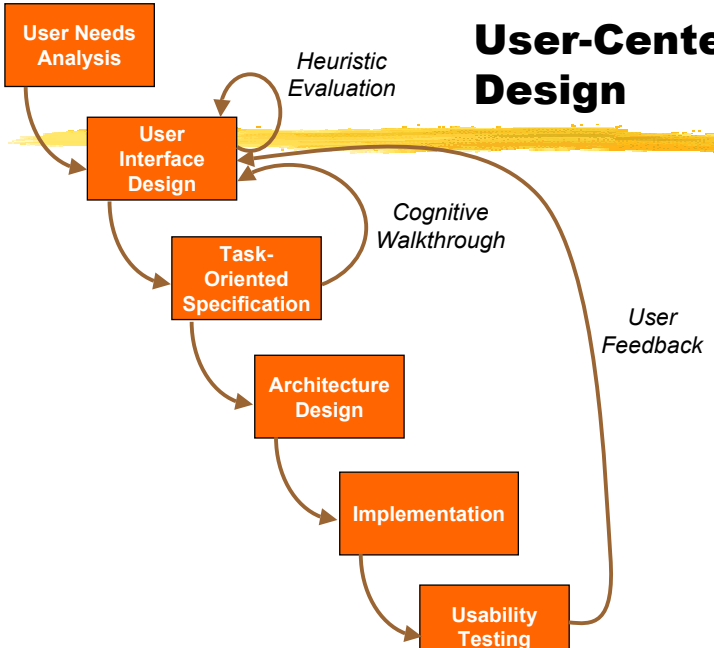
- Application Task
  - Actions completely carried out by system
  - E.g., performing some computation

- Interaction Task
  - Performed by user using system
  - E.g., clicking "back" button to go to previous web page

In-class Exercise

- Create system task model for "finding out what courses I want to take"

Software Architectures for Interactive Systems



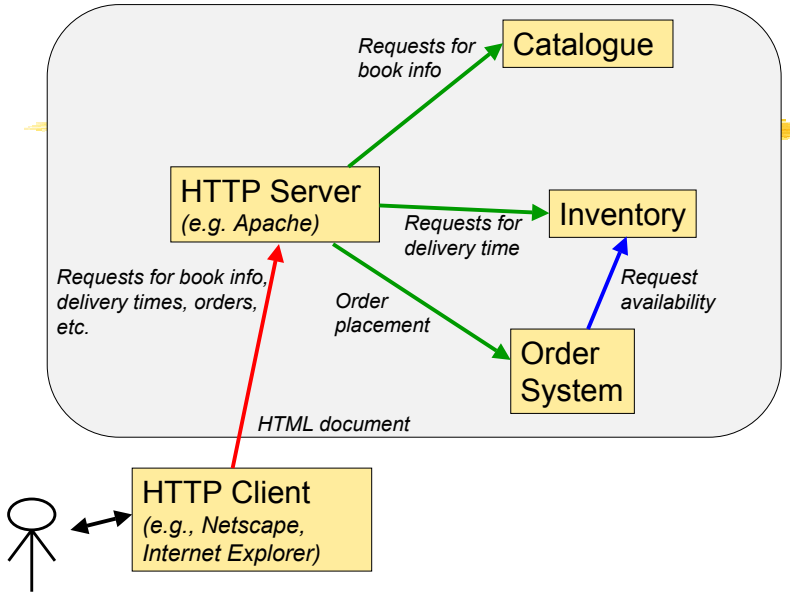
User-Centered Design

Software Architecture

- Decomposing system into set of components, connectors
  - Typically coarse grain
- Design tool
  - Helps communicate proposed structure before implementation
  - Serves as basis for dividing work packages
  - Supports analysis of implementation properties

Very Simple Software Architecture Example

- An e-commerce system supporting ordering of books on the WWW
- Customers use web browser to find books, read about books, make orders
- Inventory subsystem can estimate time before book is sent
  - Some books in inventory, some need to be ordered from publisher
- After order is placed, sent to shipping subsystem to be filled
- *For an example of such a system, see http://www.amazon.com*



Components and Connectors

Component

A software or hardware entity implementing some part of the system's functionality

Connector: a mechanism allowing components to communicate. May be a protocol, an operating system mechanism or a programming language feature. Connectors may themselves involve complex code.

Example Connectors

HTTP Connector: source component issues HTTP requests, target component responds with HTML document

CGI Connector: source component calls target component; target component responds with HTML document

Java Call Connector: source component calls method of target component using standard Java method calling conventions; target component returns some Java type

Analyzing Architectures

- Some interesting questions to ask of this architecture
  - *Availability:* What are risks of system not being available (e.g., going down?)
    - Redundant servers?
  - *Security:* What are risks of (e.g.) credit card information being intercepted
    - Perhaps use HTTPS connector instead of HTTP connector?
  - *Performance:* What are risks in case of many users?
    - Perhaps use array of servers, with server redirection?
  - *Modifiability:* What if we want to add a WAP telephone ordering service?

Architectural Styles

- Architectural design is a creative process
- Balance tradeoffs between solution properties
  - Availability, Modifiability, Security, Performance, Cost to implement, …
- When designers create a solution to an architectural problem, it is interesting to record this solution
  - Others can reuse architecture without having to experiment
  - Properties of solution well-understood
  - May help in code reuse

Architectural Styles

- An architecture style is a set of rules of how to organize a system into a set of components and connectors
- Should be accompanied by
  - *Syntax:* explanation of how to combine components and connectors
  - *Domain description:* explanation of where this style should be applied
  - *Reasoning framework:* explanation of properties of this architecture
- In practice, the reasoning frameworks for architecture styles are often poorly developed