1.0 - INTRODUCTION

In early 1994, there was a great opportunity to improve the distribution and presentation of documents at the University of Illinois at Urbana-Champaign (UIUC). A year earlier, a research laboratory at UIUC developed widely-popular tools for accessing and displaying globablly-distributed hypermedia documents. Not only did these tools greatly simplify the distribution of documents electronically, but the hypertext capabilities that the tools afforded proved to be very useful. These tools lead to an explosion of publishing on the *World-Wide Web* that has not abated.

Despite this seminal work at UIUC, there were very few electronic resources available on its campus. Instead, people had to cope with large numbers of isolated paper documents. For example, students needed three different paper-based publications to go through the registration process.

It seemed clear that moving from a paper-based documentation system to a Web-based one could make a qualitative improvement in the operations of the university. This paper describes the work that I did over three semesters towards developing UIUC Web resources. In particular, it will:

- Give a history and background of the technology behind the World-Wide Web
- Discuss the history of the Web at the University of Illinois at Urbana-Champaign
- Discuss the objectives of the project

- Explain design decisions made
- Present major sections of the UIUC Web
- Discuss the impact of the UIUC Web

My focus was on creating a framework for future work and a few exemplars of resources of wide interest. Toward that end, I created indices for UIUC resources, developed hypertext versions of registration materials, and created an integrated system of maps, floorplans, and building information.

It should be noted that I also devoted a great deal of attention to training, proselytizing, fielding questions, and giving advice. Discussions of such attention to the human side of the local Web's development, however, are outside the scope of this paper.

The Web development efforts have been extremely successful. High-level administrative officials have taken an active interest, usage by students, faculty, staff, administrators, and people from off-campus is high, and a permanent, full-time position has been created just to coordinate the university-level Web efforts.

2.0 - TECHNOLOGY HISTORY AND BACKGROUND

For years, computers have held the promise of the "paperless office", and for years, "promise" has been the operative word. Computers have instead tended to *increase* paper consumption. Electronic communications technologies - from the telegraph to electronic mail - have greatly improved the ability of people to interact with others. However, until very recently, there were very poor tools for obtaining information without the active intervention of the information provider. This is not true in the paper-based information world: anybody can go into a library or bookstore and peruse a book, walk into a car dealership and pick up promotional materials, or turn on the radio and hear an opera. This chapter will show how recent advances in technology have changed the model of electronic publishing so that a distribution system much more like the old, paper-based system was able to develop.

Until a few years ago, the only tool available on the Internet for distributing information to an unidentified audience was ftp (File Transfer Protocol). Using ftp software, people could access documents on any computer in the world, as long as the remote computer was connected to the Internet and configured with the proper software.

The ftp transactions used a very simple version of what is called a *client-server* architecture. One piece of software would be used by the person who wanted to get the information (the *client*), and another piece of software would be continuously running on the machine with the information (the *server*). To give an analogy, the server is like a butler for the computer. When a request is presented at the entrance to the computer, the server examines the request and determines if that request is understandable and can be fulfilled. If so, the server gets the information and presents it back to the requester. Thus clients can get information from the remote computer without being able to "enter" the computer themselves.

A big advantage of a client-server architecture is hardware and operating system independence. When the interface between the client software and the server software is well defined, client and server software can be written for several different types of computers. A Macintosh can use ftp client software to connect to a ftp server running on a Cray, or vice versa. A major consequence of the development of client-server architectures is that computers are no longer constrained to communicate only with other computers of the same type.

Unfortunately, finding and retrieving interesting resources with ftp was complicated. To look at a graph of stock prices, for example, might take the following steps:

- log in (using the ftp client software) to the computer with the list of stocks (the server)
- change directories
- retrieve ("download") a plain-text file
- read the file (with another program) to find out where the graph if interest is
- log off of the first computer
- log on to a second computer, the one that has the graph
- change directories
- download the graph

- view the graph (with another program)

Needless to say, the appeal of such operations was limited. In 1991, the University of Minnesota began development of a different client-server system called gopher. Gopher clients had a text-based, menu-driven, hierarchical organization of files, and allowed users to examine plain text files with the same software that they used to navigate through the menus. Furthermore, gopher provided an ability to search through all of a server's files for user-specified character strings, and allowed connecting other gopher sites to menu entries. This allowed people to aggregate information based on topic, even for information that was geographically dispersed.

However, gopher clients were unable to access ftp sites. The negotiation (called a *protocol*) that took place between the client and server was different enough that ftp sites couldn't understand gopher requests, and gopher clients didn't even allow the users to ask for information from an ftp site. To go back to the butler analogy, it was as if a courier speaking English went to the back door, while the Chinese-speaking butler was waiting at the front door. If a consumer or provider wanted to make sure of their ability to communicate with all others, they would need software for both protocols - essentially having two couriers or two butlers, one speaking English and one speaking Chinese.

Meanwhile, at Europe's CERN physics laboratory, a very small team had been working for about a year on a hypertext system, dubbed the *World-Wide Web*, for sharing research information. Tim Berners-Lee and Robert Cailliau envisioned a distributed hypertext system for publishing documents over the Internet to allow for better communication and archiving of knowledge in their organization.

CERN has an interesting environment. Not only does it have heterogeneous hardware, software, and data formats, as is true in many companies, but many researchers are there for relatively short periods. This large population of transients leads to a form of organizational amnesia. Important information was lost every time a researcher returned to his or her "home" institution.

Berners-Lee and Cailliau felt that a system to allow for easier publishing of information would help defray this continual disappearance of organizational memory. Because of the environment they were in, it was clear to them that they needed a system that would be extremely flexible, open, and extensible. The system they developed, the *World-Wide Web* had all the features of gopher in this regard, plus three more.

- Web clients (also called *browsers*) were able to request an arbitrary file in an arbitrary format from an arbitrary machine. The form of the request was a *Uniform Resource Locator*, or *URL*. Simply, a URL is an address in cyberspace. The URL compactly specified the protocol used for the transfer, the machine where the file was located, where on the machine the file was located, and finally what the file was named.

- While even the first web browsers were able to understand four existing protocols (ftp, WAIS, gopher, and NNTP), another protocol, http, was established. Among other things, http allowed for *format negotiation*. The client, when making a request, could specify the formats it was capable of dealing with. The server would select an appropriate file, perform appropriate conversions to reach a format that the browser could understand, and send it to the client along with a message specifying what format had been sent.

- A new format was defined, called *Hypertext Markup Language* or *HTML*. This language was intended to be recognizable by any Web browser, and allowed some control over the presentation of text (for example italics or bold). More importantly, it allowed the author to connect an arbitrary block of text to an arbitrary URL. When that block of text was selected - by using arrow keys, a menu, or a mouse - the client software would automatically make a request for the file at that URL, then display it. This *hyperlinking* facility was similar to gopher's ability to link to other gopher sites, but unlike gopher, allowed putting connections to other resources in the files, not just in a separate menu structure.

While format negotiation is practically unused today, that information was passed back to the client about the format meant that browsers could then start up "helper applications". A browser would not have to be capable of displaying for example chemical structure data if the browser were capable of starting another program that could display chemical structure data.

Because the specifications were an open standard, other people could and did develop WWW software. Two early browsers that came from outside CERN were `Viola` and `Cello`. `Viola` was developed by Pei Wei at Berkeley in January of 1993, and `Cello` was developed by Thomas Bruce at Cornell later in 1993. Both browsers used a page-oriented, multi-font, point-and-click interface.

Meanwhile, Marc Andreessen and Eric Bina at the University of Illinois' National

Center for Supercomputing Applications (NCSA) developed a browser called Mosaic. This browser had point-and-click capabilities, like Viola and Cello, but it could also display graphics in-line. No helper program was needed to view graphics. This feature, good timing, and the resources available to develop, publish, and support binary, ready-to-use versions for X Window, Mac, and PC clients made Mosaic far more popular than any of its contemporaries.

Rob McCool, who was also with NCSA, developed a web server called httpd. Collaborating with the Mosaic team, McCool developed a mechanism that allowed a server to execute a program and send its outputs to the browser as if it were a static file. This *Common Gateway Interface* (*CGI*) allowed dynamic data to be transmitted over the web. Things as mundane as the local time in various places around the world, and as exotic as an interactive map of the world with arbitrary scale were placed onto the web using CGI programs. Kevin Hughes, who was a student at Honolulu Community College, wrote an important CGI program that allowed for "clickable" images. Regions could be defined inside an inline image; clicking in such an area would take the user to another URL. This was incorporated into NCSA's httpd distribution.

By this point, the Web had all the components needed for a universal user interface for unknown-audience document publishing. Independence of physical location, hardware, operating system, software, and transfer protocol for both servers and clients, and format independence for the data meant that practically anybody could easily publish and/or view information on the Web.

Furthermore, the Web being an open and extensible system meant that obsolesence would not be an issue for some time, if ever. New protocols and new data formats might spring up, but they would augment, not eliminate earlier versions. This, and the easy availability of free browsers, servers, and helper applications that were easy to install and use lead to a remarkable growth around the world in the Web.

3.0 - HISTORY OF THE UIUC WEB

Despite so much of the early Web development happening at the University of Illinois' National Center for Supercomputing Applications, the amount of material on the Web about the university was surprisingly small andits organization ad hoc. This chapter will describe the state of the UIUC Web prior to this project, discuss barriers to campus Web development, and show the early evolution of this project.

By the spring of 1994, NCSA had developed web pages for their organization and a rich site for the Krannert Art Museum (mostly to demonstrate the graphic capabilities of Mosaic). To have some sort of university-level information for their demonstration site, NCSA had requested and received a publicity brochure from the Office of Public Affairs, and translated it into hypertext. This brochure was long on photos, but had only the most superficial information about the university.

The Computing and Communications Services Office (CCSO) had a server on-line with a web page for university-level information. However, the developer of that page, Ed Krol, was on leave at that time to promote his best-selling book, *The Whole Internet User's Guide and Catalog*. All that existed on the page was an image that changed automatically according to the current weather conditions, linked to the local weather forecast. This was technically a neat trick (requiring intelligent parsing of the local weather forecast), but it was limited in its utility.

There were also some UIUC sites with reasonable information that was restricted in scope. For example, Ed Kubaitis of CCSO had developed a rich set of indices to information available (mostly elsewhere on the web) of interest to the College of

Engineering. The Departments of Electrical and Computer Engineering and Computer Science, the College of Agriculture and the Graduate College, and the student societies of Eta Kappa Nu and Association for Computing Machinery had pages available.

While there was a list of all the servers on campus, it was organized from a technology point of view - what *computers* hosted information instead of what *information* was provided. The sites also varied greatly in the amount and type of information served. The College of Agriculture, for example, used their web site mostly for training (for example how to use electronic mail) while the Department of Computer Science used its site to present reference material, centered around their advising handbook.

It was somewhat surprising how ad hoc and scattered the UIUC Web was. I believe that one reason why the UIUC Web was underpopulated was *because* so much Web development was happening at NCSA. Energetic computer science students who wanted to do something with the neat new technology were quickly hired by NCSA to work on Web tools, instead of playing on their own to develop Web content.

Another problem that was not unique to UIUC was that this technology landed in a gap between traditional departmental responsibilities. CCSO had the technical skills needed to publish on the Web, but were not chartered to do content development. The Office of Public Affairs and the Office of Publications develop documents for external and internal consumption, respectively, but had no experience with the Internet. To complicate matters, this medium needed to serve both internal and external consumers of information.

Furthermore, while at that time the technical underpinnings of the Web were complete, and Vice-President Al Gore had popularized the phrase "information superhighway", very few people had ever heard of the Internet. Even fewer had heard of the World-Wide Web. Of those who had heard of the Web, a significant number were certain that it would be at worst a mere fad, or at best quickly replaced by something more technologically advanced. The unfulfilled promises of multimedia and "the paperless office" had jaded many into a "wait-and-see" stance.

It was into this void that I jumped. I first developed pages for what was then my research group, the Illinois Genetic Algorithm Laboratory (IlliGAL). But what really got me started on university-level information was registering for classes in April of 1994. I got extremely frustrated at having to move back and forth between the departmental graduation requirements, the course descriptions, and the schedule of when and where classes were offered. I recognized that this was a perfect application of hypertext, so I developed a Web version of the *Courses Catalog* and *Timetable*, first for General Engineering classes and then for all classes in the College of Engineering and the Mathematics Department.

Because there wasn't a page for the General Engineering department where the lab's page and the registration material could be listed, I created a page for the General Engineering department. Because there wasn't a good place to list the General Engineering department or the college-level class information, I created a page for the College of Engineering.

At this point, I notified CCSO that the pages existed, and asked if they could perchance find some disk space available to me, as I was running close to my quota on my student account. This lead to my being offered, the next semester, an assistantship as Webmaster for the university with absolute freedom in topics and implementation in what resources to develop.

To summarize, the university Web was hampered by a drain of talent to NCSA, ignorance of the Web, unclear jurisdiction, and technical inexperience. There were some good resources, but development was at a very low organizational level. Resources were laboratory-, department-, or occasionally college-based, with no university-level resources. There was not a good, subject-based index to university resources. This was the state of the UIUC Web when I was given the explicitly broad directive, "Go do great things."

4.0 - OBJECTIVES

I chose to interpret "great things" to mean "elimination of all paper documents at the University of Illinois at Urbana-Champaign". However, since it clearly would not be possible for one person to develop hypermedia resources for all aspects of a 35,000-student university, and as I wished to graduate in the twentieth century, I recognized that I would need to restrict my enthusiasm somewhat and work towards encouraging other people to add to the body of work. This chapter will discuss the goals I kept and why.

Developing a well-developed framework was a high priority. Not only would a good, centralized index of university resources be useful to people seeking information, but done properly, would encourage people to develop resources. Any comprehensive list of organizational units - with only the units providing web pages hyperlinked - would make immediately obvious the groups without web resources. I felt this might goad individuals from laggard organizations to push for development of their own web resources.

It was also clear that some significant resources should be developed to pull people into using the Web, and to act as exemplars of what could be done. I felt that there were several important features for exemplars. They should be items of wide interest and provide a better presentation of the information than the paper version. If the items were visually interesting and/or unavailable on paper, this would also encourage Web usage and thus development.

The resources that I decided to focus on pertained to the registration information that I had already started and those pertaining to navigational aids. Almost everyone on the campus is involved in some way with classes. As for the navigational aids, the University of Illinois is a geographically very large institution, with ample territory for getting lost. While some navigational resources are available -- for example, a map printed in all the phone books -- more detailed and sophisticated navigational resources were simply not available. Navigational systems have the added benefit of being visually appealing.

This was the motivation behind focusing on my three main objectives:

- Develop a framework for integration of others' work
- Develop course and program information
- Develop navigational aids

I felt that these resources would provide a good foundation for the UIUC Web. The framework would allow people to find information and stimulate further work. The registration materials would be of immediate, wide-spread use. The navigational system would be useful and also expand people’s awareness of the capabilities of Web technology.

5.0 - METHODOLOGY

Because so little precedent had been set on the Web, the procedure for developing documents was of necessity quite experiential. I learned what I could from other web sites, developed pages and code, and revised them based on my own observations and the copious feedback received.   This chapter will discuss the guiding principals that I used.

One thing to note is that the Web is very forgiving. Unlike paper documentation, the overhead associated with releasing a new version of the information is negligible. Furthermore, electronic feedback is easy and immediate. This tighter bond between publishers and readers means that if there is an error on a page, the maintainer *will* find out about it.

5.1 - Implementation Decisions

`perl`

For the sections of the UIUC Web that were dynamically generated (mostly the navigation system), the programming language `perl` was chosen as the implementation language. `Perl` is a scripting language particularly well suited to text manipulation. It is an interpreted language, and so is somewhat slower than a compiled language like `C`. However, at the commencement of the project, the demands on the computer running the web server were so minimal that this did not appear to be an issue. Furthermore, many of the scripts would only execute infrequently, with the results of the execution being stored for common retrieval.

`Mosaic` Compatible

The two most popular graphical browsers available were `Mosaic` and `Netscape`, a `Mosaic` spin-off. `Netscape` supports extensions to HTML that did not originate with the official standards body, and that are not recognized by `Mosaic`. In many places, this has caused no little soul-searching as people grappled with the decision of whether to use the Netscape extensions. Because `Mosaic` was developed at the University of Illinois, I had no such dilemma. The resources that I developed used no extensions specific to `Netscape`.

5.2 - Design Principles

Conversion Not Creation

Given that by this point in the mid-90's, almost all documents start their lives on hard disks somewhere, and given the number of documents that exist, it was logical to decide to focus on converting existing documents instead of generating new documents, scanning documents using optical character recognition, or re-typing the text of old documents.

Text Over Graphics

For the early work, I used no decorative graphics. I knew that there were many users on campus who did not have particularly fast machines, and that many people from off-campus (and a few on-campus) accessed the Web using relatively slow modems. I thus made the conscious decision to avoid graphics unless the graphic was the item of interest, as is true for e.g. a map.

As time went on, and the site became more formal, decorative graphics were essentially mandated by the Office of Public Affairs. While I resisted initially, I now recognize that decorative graphics do add value: they are a form of conspicuous consumption that shows that an organization has adequate resources to be able to devote some to artwork. Still, I fought to keep the artwork small and to make sure that users could still navigate adequately if they set their browser such that images would not be downloaded.

Strongly Cross-Linked

It would have been easy to put information on-line in its plain-text form. The information usually came from the providers in that form. However, raw text is less legible and hence less usable than the equivalent paper version. Related paper documents could instead be converted into cross-linked documents on-line, i.e. with hyperlinks connecting the two documents tightly. The ease of shifting between documents is thus a strong value added to the original text. Thus emphasis was placed more on making highly cross-linked documents than on sheer volume of documents.

Regular Addressing

To make it easy for subsequent users to link to documents, the Uniform Resource Locators (URLs) for documents were made as regular as possible. The URLs were in fact so regular that in some cases, it was possible to automatically create hyperlinks in raw text via a `perl` script.

Instrument Code

It was relatively simple to instrument the code so that if an error condition was detected, the program sent email to me with pertinent information. This gave far more and better diagnostic information than users tend to give.

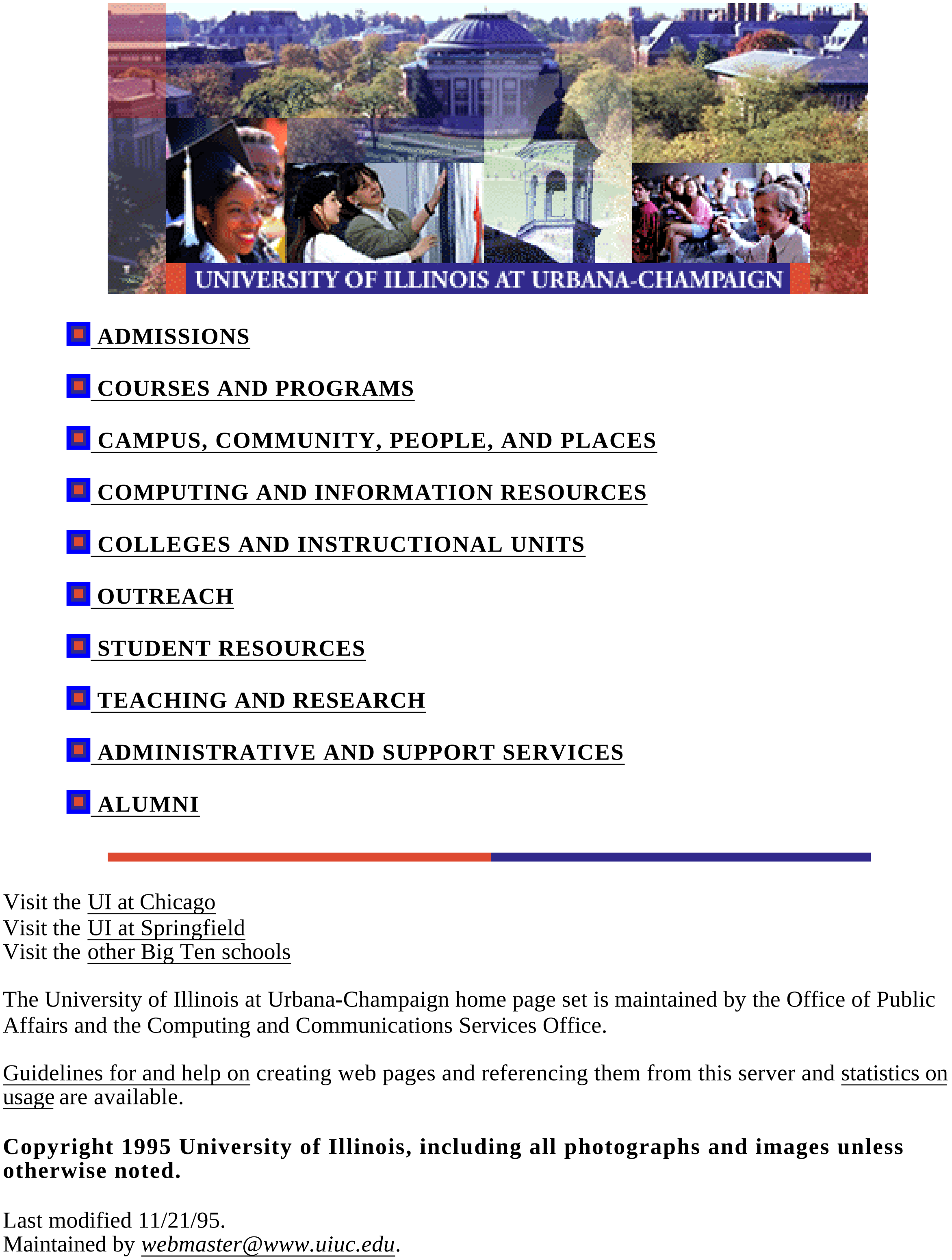
Because the Web is so new, these heuristics were based soley on my experience as a Web user. As the Web matures, there will certainly be more codification of good design principles.

6.0 - FRAMEWORK OF THE UIUC WEB

It was important to create indices to information of use and interest to the campus community. Being able to find information easily would lead to more use of the Web, which I presumed would lead to more interest in publishing information. This chapter will show how this framework was built.

At first, almost everything was linked from the top page in the hierarchy, the university's *home page*. As the local Web grew, however, it was necessary to expand the indices (with the help of the Publications Department) to more levels. The UIUC home page can be seen in Figure 1; pages of the second level can be seen in Figures 2 through 9. An important third-level page, the *References* page, can be seen in Figure 10. Underlined words and phrases are hypertext links, and will take the user to further information on that topic.

There are two interesting things to mention. The first is that not everything was hyperlinked initially. In particular, the *Colleges and Instructional Units* page had links to only about a quarter of the units (with two more added quickly only because I myself developed skeletal home pages for those colleges in order to be able to "plug in" departments in those colleges).



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

- **ADMISSIONS**
- **COURSES AND PROGRAMS**
- **CAMPUS, COMMUNITY, PEOPLE, AND PLACES**
- **COMPUTING AND INFORMATION RESOURCES**
- **COLLEGES AND INSTRUCTIONAL UNITS**
- **OUTREACH**
- **STUDENT RESOURCES**
- **TEACHING AND RESEARCH**
- **ADMINISTRATIVE AND SUPPORT SERVICES**
- **ALUMNI**

Visit the UI at Chicago
Visit the UI at Springfield
Visit the other Big Ten schools

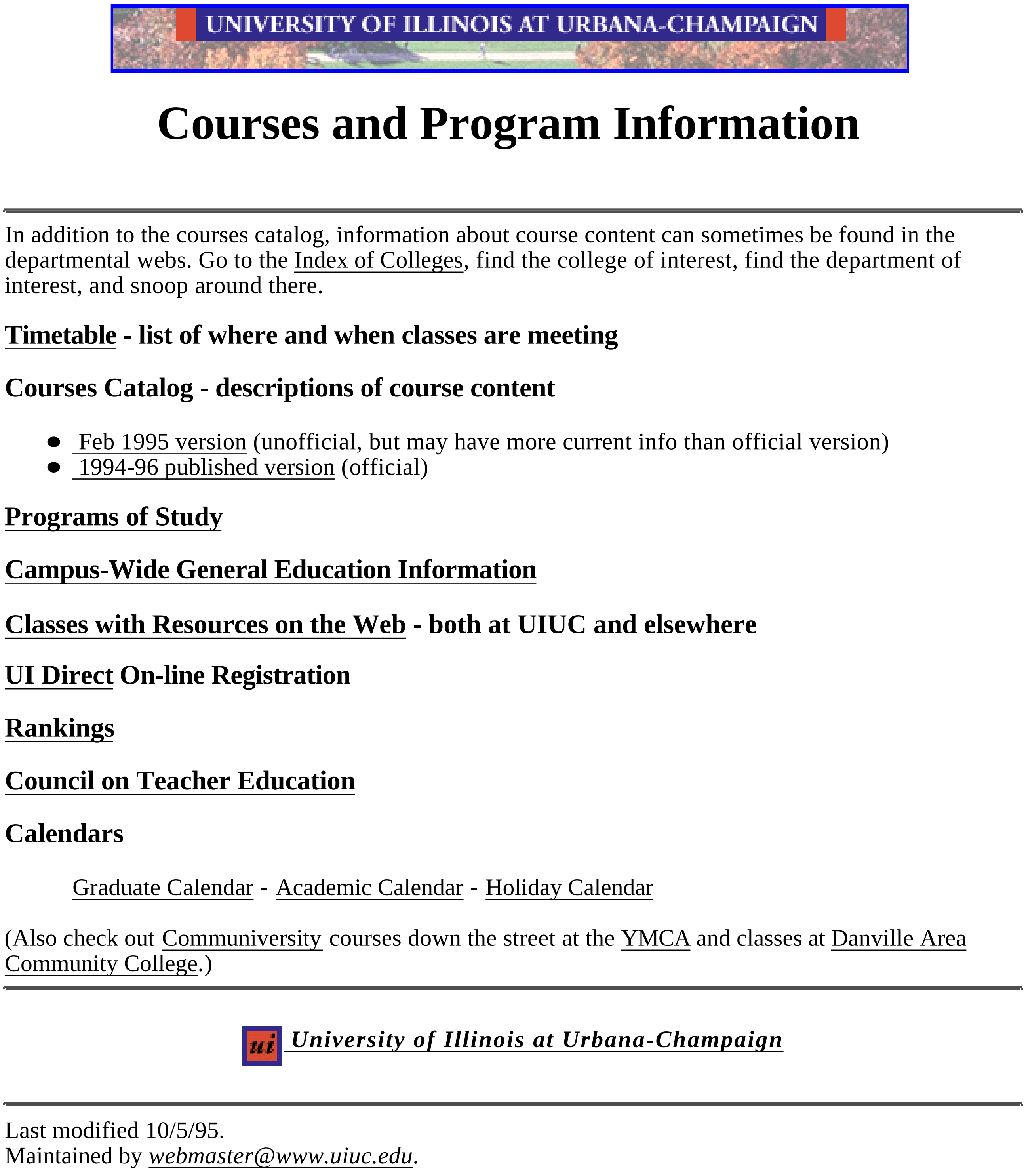
The University of Illinois at Urbana-Champaign home page set is maintained by the Office of Public Affairs and the Computing and Communications Services Office.

Guidelines for and help on creating web pages and referencing them from this server and statistics on usage are available.

Copyright 1995 University of Illinois, including all photographs and images unless otherwise noted.

Last modified 11/21/95.
Maintained by *webmaster@www.uiuc.edu*.

Figure 1 - UIUC Home Page



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Courses and Program Information

In addition to the courses catalog, information about course content can sometimes be found in the departmental webs. Go to the Index of Colleges, find the college of interest, find the department of interest, and snoop around there.

Timetable - list of where and when classes are meeting

Courses Catalog - descriptions of course content

- Feb 1995 version (unofficial, but may have more current info than official version)
- 1994-96 published version (official)

Programs of Study

Campus-Wide General Education Information

Classes with Resources on the Web - both at UIUC and elsewhere

UI Direct On-line Registration

Rankings

Council on Teacher Education

Calendars

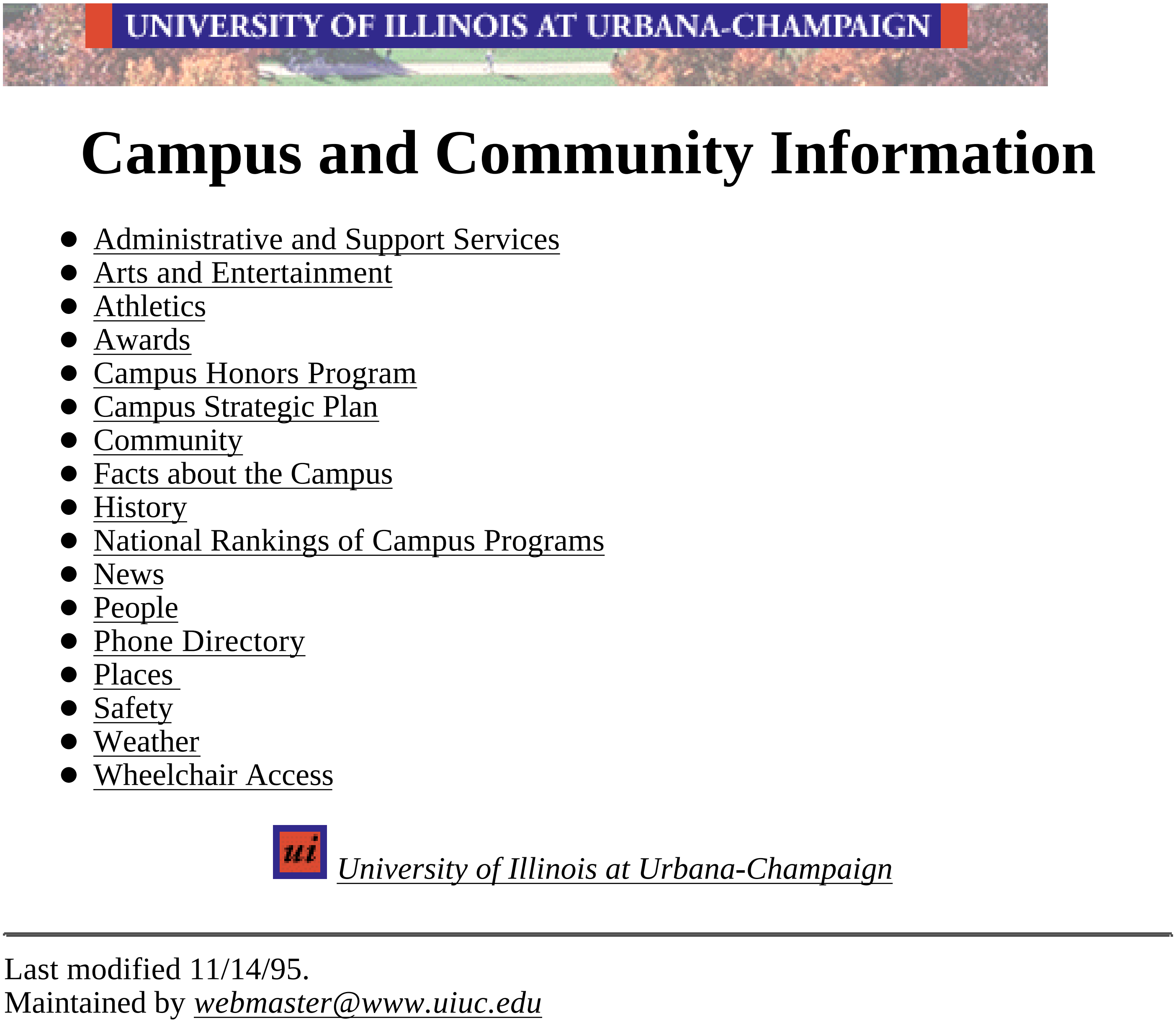
Graduate Calendar - Academic Calendar - Holiday Calendar

(Also check out Communiversity courses down the street at the YMCA and classes at Danville Area Community College.)

University of Illinois at Urbana-Champaign

Last modified 10/5/95.
Maintained by *webmaster@www.uiuc.edu*.

Figure 2 - Course and Program Information



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

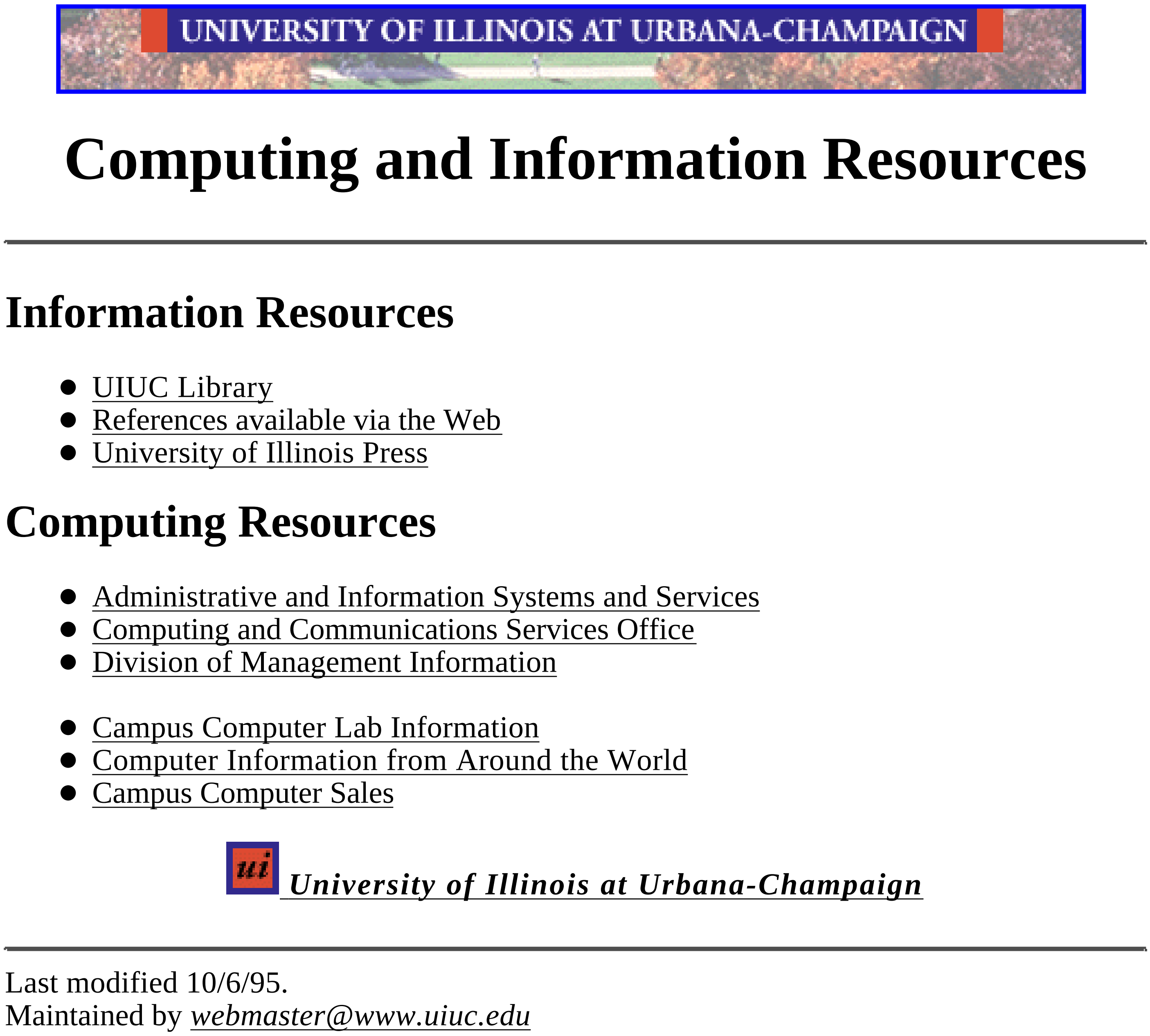
Campus and Community Information

- Administrative and Support Services
- Arts and Entertainment
- Athletics
- Awards
- Campus Honors Program
- Campus Strategic Plan
- Community
- Facts about the Campus
- History
- National Rankings of Campus Programs
- News
- People
- Phone Directory
- Places
- Safety
- Weather
- Wheelchair Access

University of Illinois at Urbana-Champaign

Last modified 11/14/95.
Maintained by *webmaster@www.uiuc.edu*

Figure 3 - Campus and Community Information Page



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Computing and Information Resources

Information Resources

- UIUC Library
- References available via the Web
- University of Illinois Press

Computing Resources

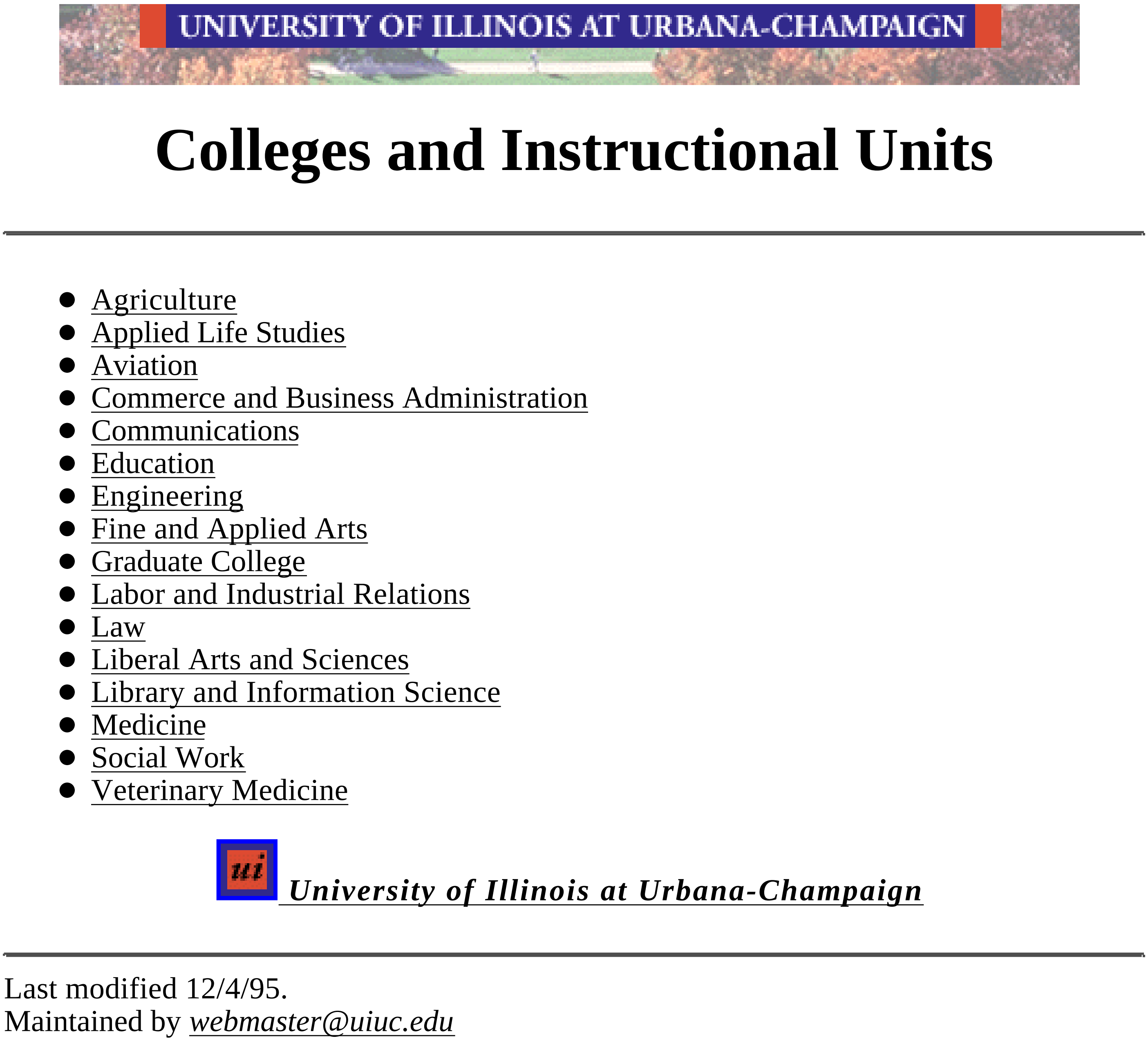
- Administrative and Information Systems and Services
- Computing and Communications Services Office
- Division of Management Information

- Campus Computer Lab Information
- Computer Information from Around the World
- Campus Computer Sales

ui ***University of Illinois at Urbana-Champaign***

Last modified 10/6/95.
Maintained by *webmaster@www.uiuc.edu*

Figure 4 - Computing and Information Resources Page



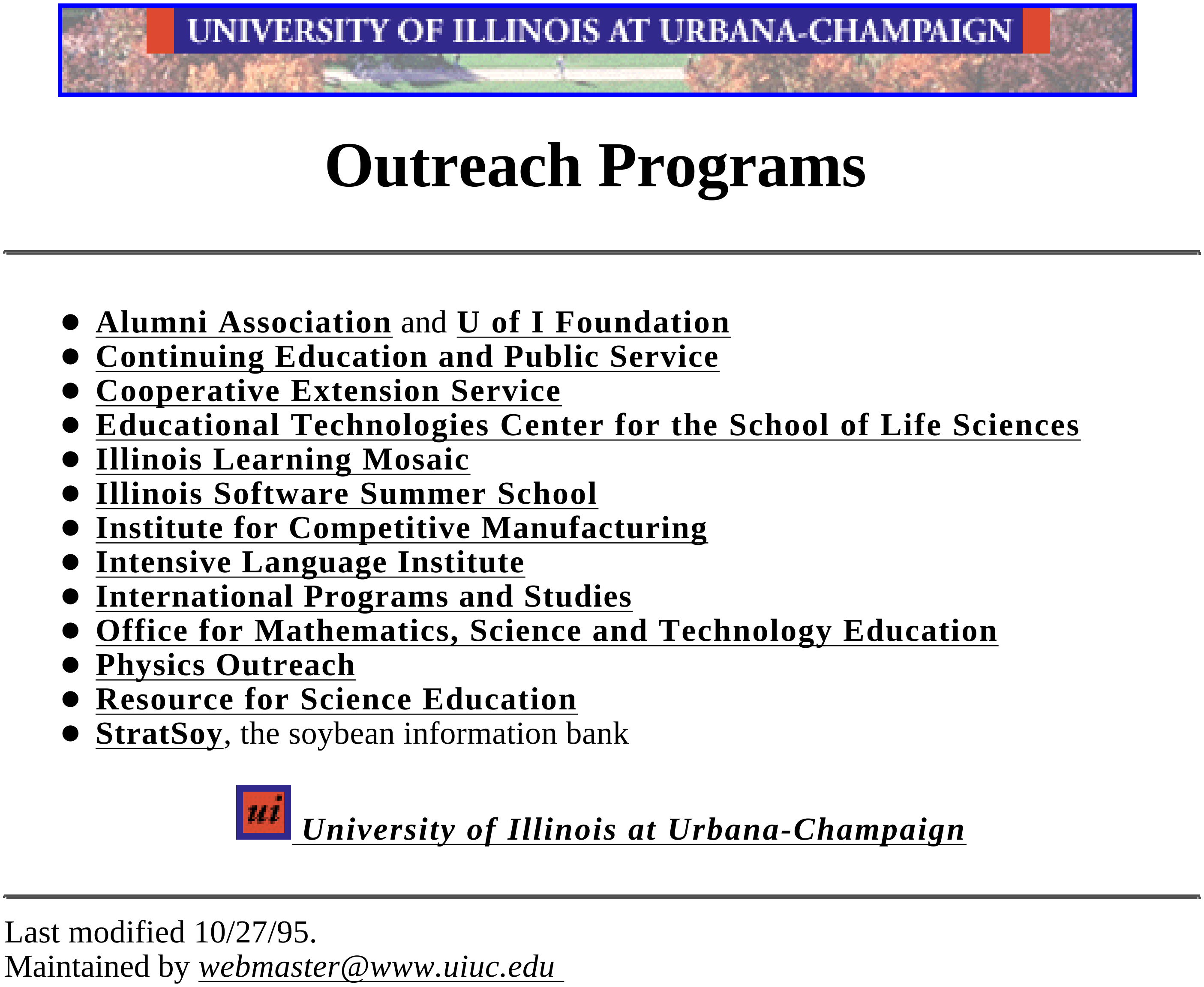
Colleges and Instructional Units

- Agriculture
- Applied Life Studies
- Aviation
- Commerce and Business Administration
- Communications
- Education
- Engineering
- Fine and Applied Arts
- Graduate College
- Labor and Industrial Relations
- Law
- Liberal Arts and Sciences
- Library and Information Science
- Medicine
- Social Work
- Veterinary Medicine

University of Illinois at Urbana-Champaign

Last modified 12/4/95.
Maintained by *webmaster@uiuc.edu*

Figure 5 - Colleges and Instructional Units Page



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

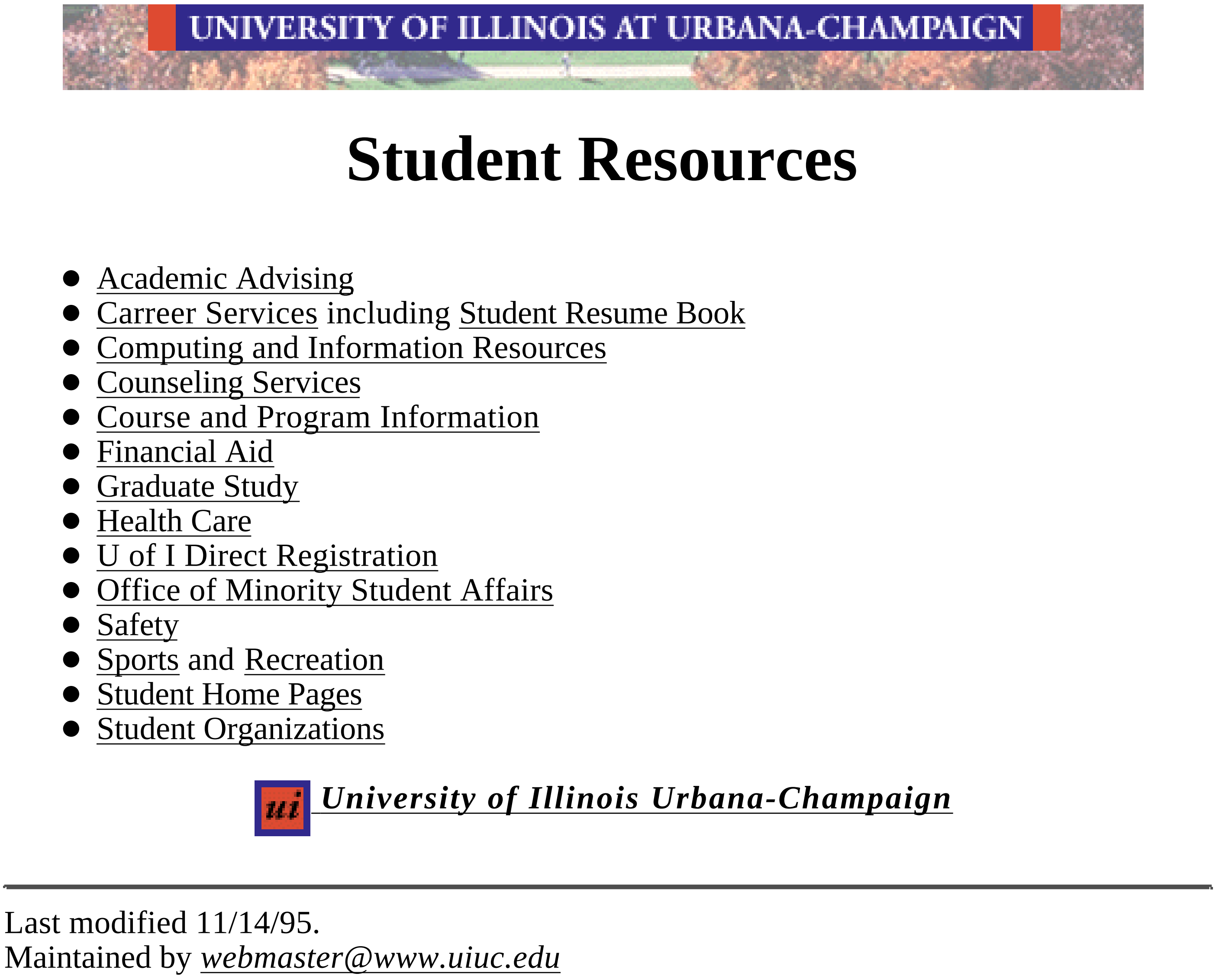
Outreach Programs

- Alumni Association and U of I Foundation
- Continuing Education and Public Service
- Cooperative Extension Service
- Educational Technologies Center for the School of Life Sciences
- Illinois Learning Mosaic
- Illinois Software Summer School
- Institute for Competitive Manufacturing
- Intensive Language Institute
- International Programs and Studies
- Office for Mathematics, Science and Technology Education
- Physics Outreach
- Resource for Science Education
- StratSoy, the soybean information bank

University of Illinois at Urbana-Champaign

Last modified 10/27/95.
Maintained by *webmaster@www.uiuc.edu*

Figure 6 - Outreach Programs Page



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

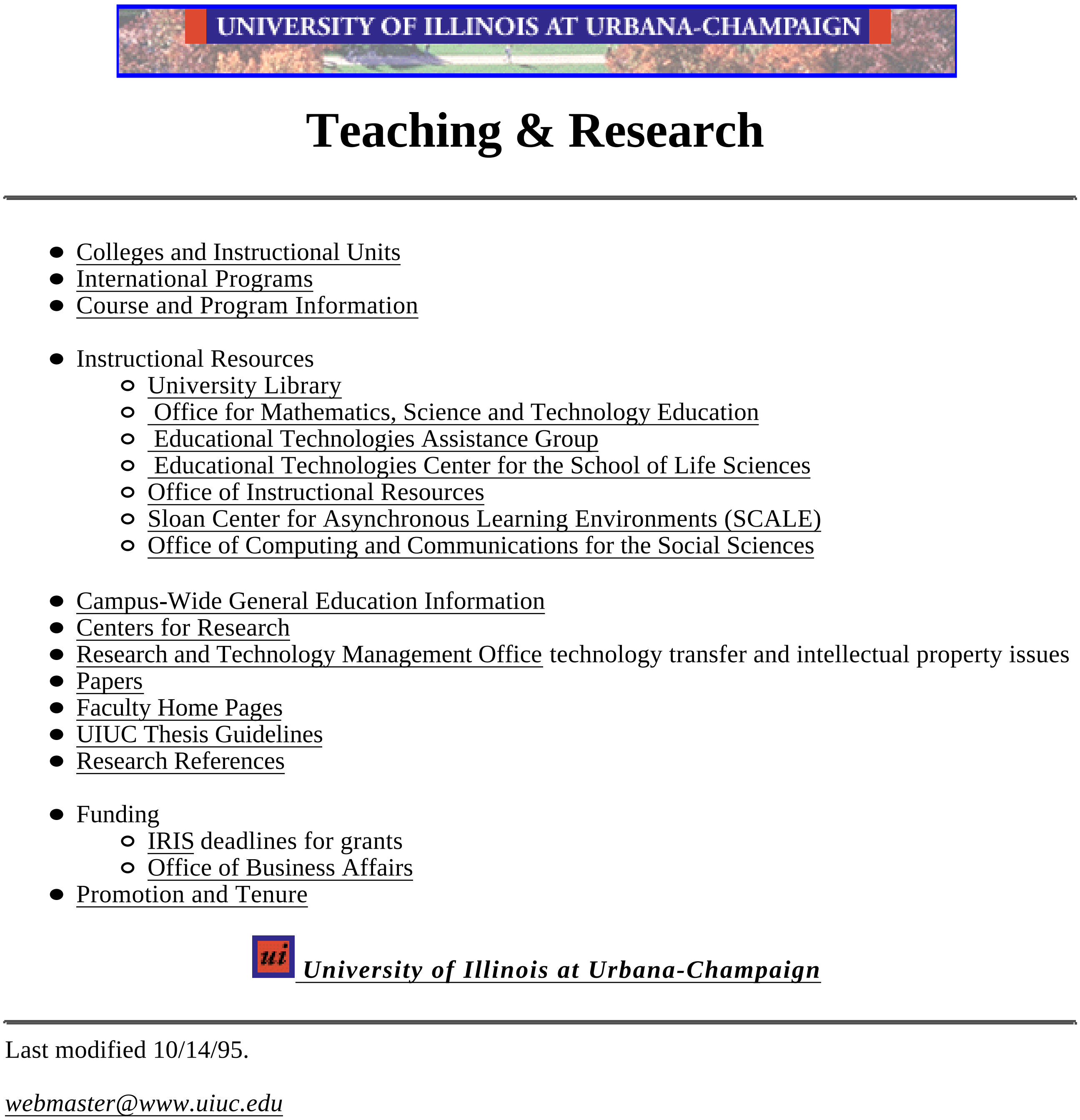
Student Resources

- Academic Advising
- Carreer Services including Student Resume Book
- Computing and Information Resources
- Counseling Services
- Course and Program Information
- Financial Aid
- Graduate Study
- Health Care
- U of I Direct Registration
- Office of Minority Student Affairs
- Safety
- Sports and Recreation
- Student Home Pages
- Student Organizations

ui *University of Illinois Urbana-Champaign*

Last modified 11/14/95.
Maintained by *webmaster@www.uiuc.edu*

Figure 7 - Student Resources Page



UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Teaching & Research

- Colleges and Instructional Units
- International Programs
- Course and Program Information

- Instructional Resources
  - University Library
  - Office for Mathematics, Science and Technology Education
  - Educational Technologies Assistance Group
  - Educational Technologies Center for the School of Life Sciences
  - Office of Instructional Resources
  - Sloan Center for Asynchronous Learning Environments (SCALE)
  - Office of Computing and Communications for the Social Sciences

- Campus-Wide General Education Information
- Centers for Research
- Research and Technology Management Office technology transfer and intellectual property issues
- Papers
- Faculty Home Pages
- UIUC Thesis Guidelines
- Research References

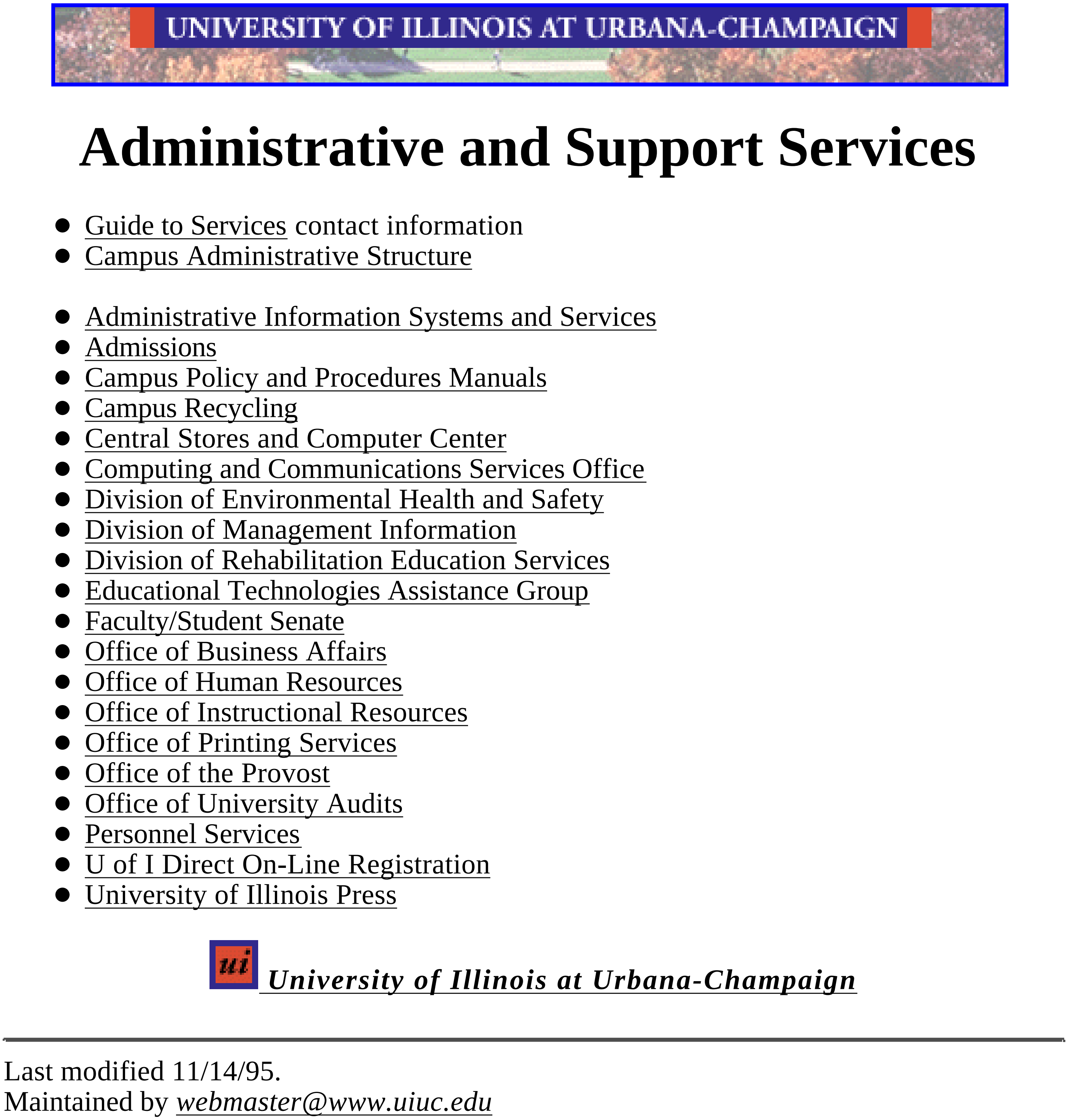
- Funding
  - IRIS deadlines for grants
  - Office of Business Affairs
- Promotion and Tenure

University of Illinois at Urbana-Champaign

Last modified 10/14/95.

webmaster@www.uiuc.edu

Figure 8 - Teaching and Research Page



Administrative and Support Services

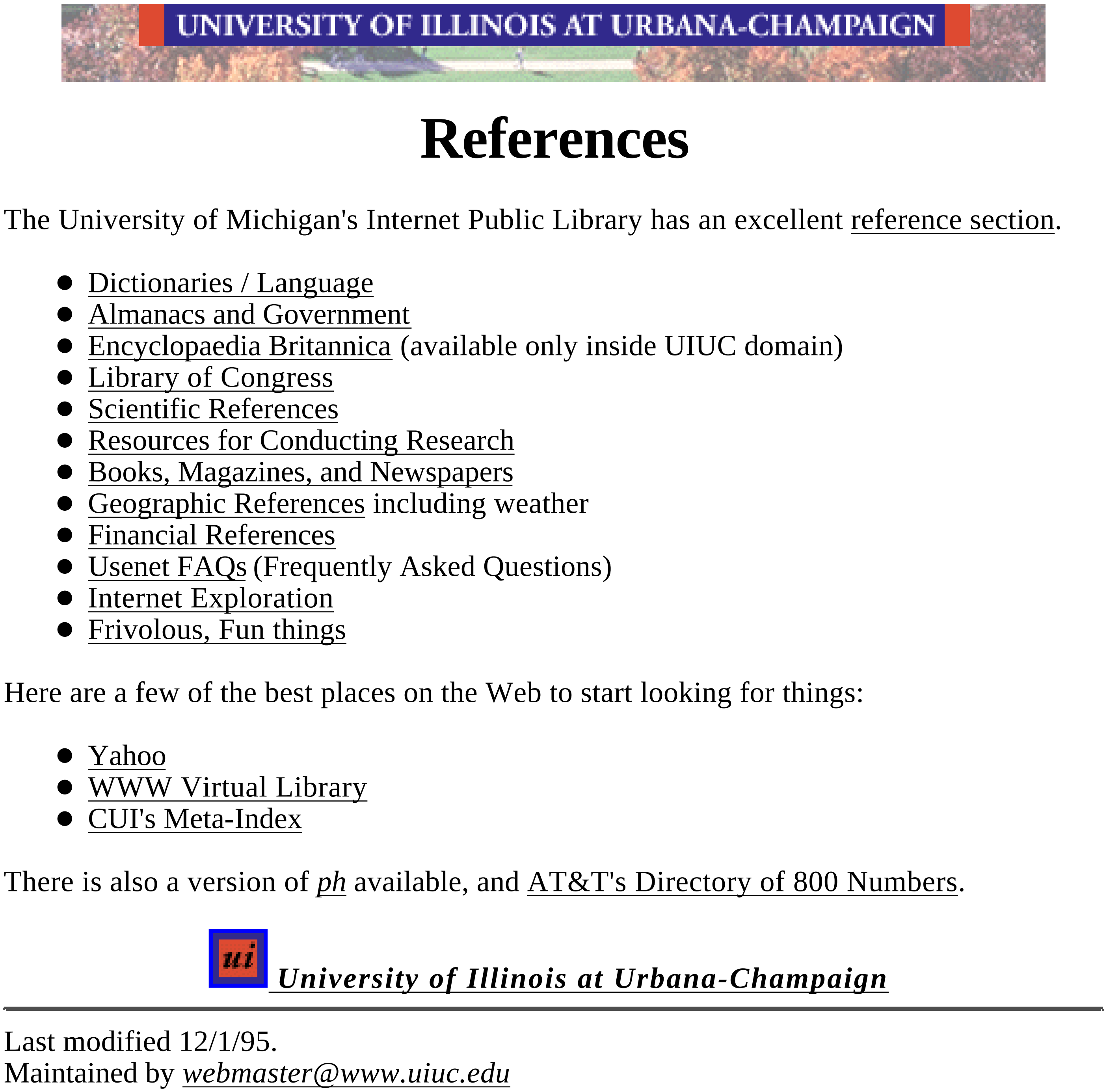
- Guide to Services contact information
- Campus Administrative Structure

- Administrative Information Systems and Services
- Admissions
- Campus Policy and Procedures Manuals
- Campus Recycling
- Central Stores and Computer Center
- Computing and Communications Services Office
- Division of Environmental Health and Safety
- Division of Management Information
- Division of Rehabilitation Education Services
- Educational Technologies Assistance Group
- Faculty/Student Senate
- Office of Business Affairs
- Office of Human Resources
- Office of Instructional Resources
- Office of Printing Services
- Office of the Provost
- Office of University Audits
- Personnel Services
- U of I Direct On-Line Registration
- University of Illinois Press

ui ***University of Illinois at Urbana-Champaign***

Last modified 11/14/95.
Maintained by *webmaster@www.uiuc.edu*

Figure 9 - Administrative and Support Services Page



References

The University of Michigan's Internet Public Library has an excellent reference section.

- Dictionaries / Language
- Almanacs and Government
- Encyclopaedia Britannica (available only inside UIUC domain)
- Library of Congress
- Scientific References
- Resources for Conducting Research
- Books, Magazines, and Newspapers
- Geographic References including weather
- Financial References
- Usenet FAQs (Frequently Asked Questions)
- Internet Exploration
- Frivolous, Fun things

Here are a few of the best places on the Web to start looking for things:

- Yahoo
- WWW Virtual Library
- CUI's Meta-Index

There is also a version of *ph* available, and AT&T's Directory of 800 Numbers.

University of Illinois at Urbana-Champaign

Last modified 12/1/95.
Maintained by *webmaster@www.uiuc.edu*

Figure 10 - References Page

The other point to note is that is that many pages that are now at the third level have been moved down. These pages are primarily ones that deal with either entertainment or references, and consist mostly of links to resources outside the UIUC Web. Initially, because the UIUC Web was so sparse, outside links and/or links to things outside the main function of the university were more prominent. In particular, the *References* page used to be very prominent, and a fair amount of effort was taken to develop the sub-categories of references. More local, relevant content displaced those pages.

The transition to more layers of hierarchy was not without pain. One of the more challenging aspects of my job after the restructuring was tactfully explaining to people why their Web site didn't rate a mention at the top level of hierarchy. There was also a conflict between keeping the page length short and providing enough information that people could figure out what was at lower levels.

In addition to the pages in Figures 1 through 9 and the sub-pages on the *References* page (language, government, scientific, research, periodicals, geographic, financial, and fun), I developed indices to Big Ten schools, classes with Web resources, arts and entertainment, news, people, career information, sports and recreation, research laboratories, and research papers.

Furthermore, I developed a number of skeletal pages. These were usually done for personal reasons - because I myself had an affiliation or need. For example, I had a BS in Metallurgical Engineering and a need for a job, which lead me to develop the Materials Science and Engineering home page and an on-line resume book.

Sometimes, however, I would create a page out of a need for a "bridge" between information that became available and the existing indices.

For these reasons, I developed the original pages for: the departments of General Engineering, Mechanical and Industrial Engineering, and Materials Science and Engineering, the Colleges of Engineering, Liberal Arts and Sciences, and Fine and Applied Arts, the Illinois Genetic Algorithms Laboratory, the Dean's Student Advisory Committee, the Society of Women Engineers, International Programs in Engineering, the Engineering Placement Office, awards, the Engineering Handbook, the Guide to Services, campus computer laboratories, and syllabi for several of the courses I took.

These were not intended to be more than placeholders, there until the appropriate information provider could take them over. By the time I left campus, all but six (not counting the syllabi) had been expanded and updated by the responsible group.

This chapter showed the organizational structure of the UIUC Web’s indicies. Those indices were not adequate, however, and in many cases pages had to be developed to fill holes in the logical structure.

7.0 - NAVIGATIONAL SYSTEM

Because the Web is able to deliver dynamic, graphic information, navigational information need no longer be an ad hoc affair, given in words by people whose spatial memory comes with no guarantees of quality. Most people still navigate by reading notes that they scrawled while on the phone, which seem to invariably leave out or get wrong at least one crucial piece of information. The directions won't note that the southeastern staircase must be used, or that a key is required after five o'clock, or that the building is not visible from any street.

My goal was to provide a usable navigation system, one that allowed people to not only easily find information that would help people navigate through the campus but to easily link to pages that would show others how to get to the location of interest. Thus people would be encouraged to make links to maps on their home pages showing the location of their office, seminar announcements, meeting locations, and classroom locations.

Towards that end, I built a navigational system consisting of many tightly interlocking pieces:

- building information
- floorplans and maps
- a "virtual walkthrough"
- wheelchair access information

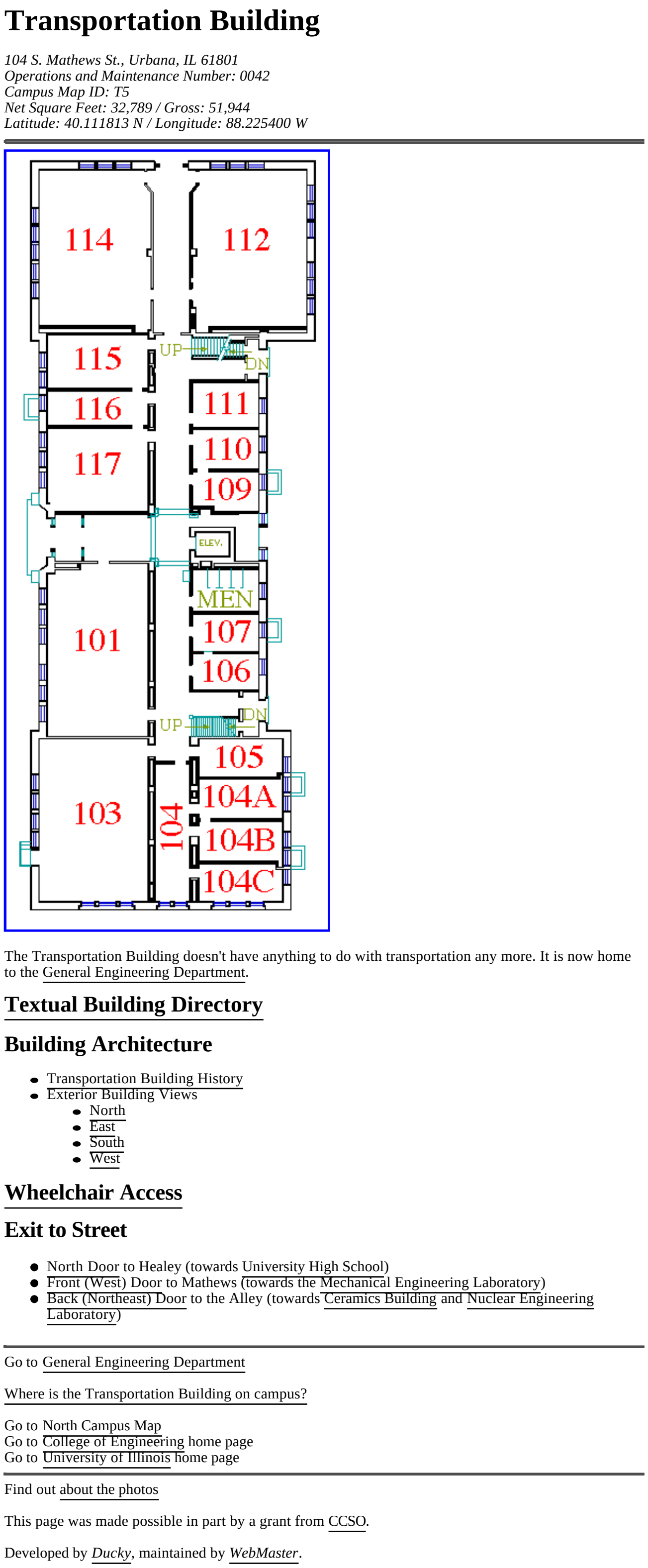
This chapter will describe these pieces and how they link together from a user's perspective and how they were implemented.

7.1 - Navigational System Features

Building Information

134 of the campus buildings now have information on the Web. What information exists varies, but all buildings have information on their street address, Operations and Maintenance identification number, and full name. Most of the more prominent buildings have a list of the organizations housed in the building (hyperlinked if the organizations have Web pages) and a photo of the building. Twenty-nine buildings have latitude and longitude coordinates for the building. Two buildings have information on their architectural history, twenty-four have wheelchair access information, and seventeen have floorplans available. For an example of building information, see the Transportation Building information page (or *lobby*) in Figure 11.

People used to ask me why I included latitude and longitude information. I used to say, "I don't know, but if I put it there, someone might find a novel use for it. If I don't put it there, nobody will find a novel use for it." There is now some talk, however, of using that information to estimate travel time between points. This travel time could be used to advise registering students of overly aggressive transfers between classes.



Transportation Building

104 S. Mathews St., Urbana, IL 61801
Operations and Maintenance Number: 0042
Campus Map ID: T5
Net Square Feet: 32,789 / Gross: 51,944
Latitude: 40.111813 N / Longitude: 88.225400 W

The Transportation Building doesn't have anything to do with transportation any more. It is now home to the General Engineering Department.

Textual Building Directory

Building Architecture

- Transportation Building History
- Exterior Building Views
  - North
  - East
  - South
  - West

Wheelchair Access

Exit to Street

- North Door to Healey (towards University High School)
- Front (West) Door to Mathews (towards the Mechanical Engineering Laboratory)
- Back (Northeast) Door to the Alley (towards Ceramics Building and Nuclear Engineering Laboratory)

Go to General Engineering Department

Where is the Transportation Building on campus?

Go to North Campus Map
Go to College of Engineering home page
Go to University of Illinois home page

Find out about the photos

This page was made possible in part by a grant from CCSO.

Developed by *Ducky*, maintained by *WebMaster*.

Figure 11 - Building Lobby Example

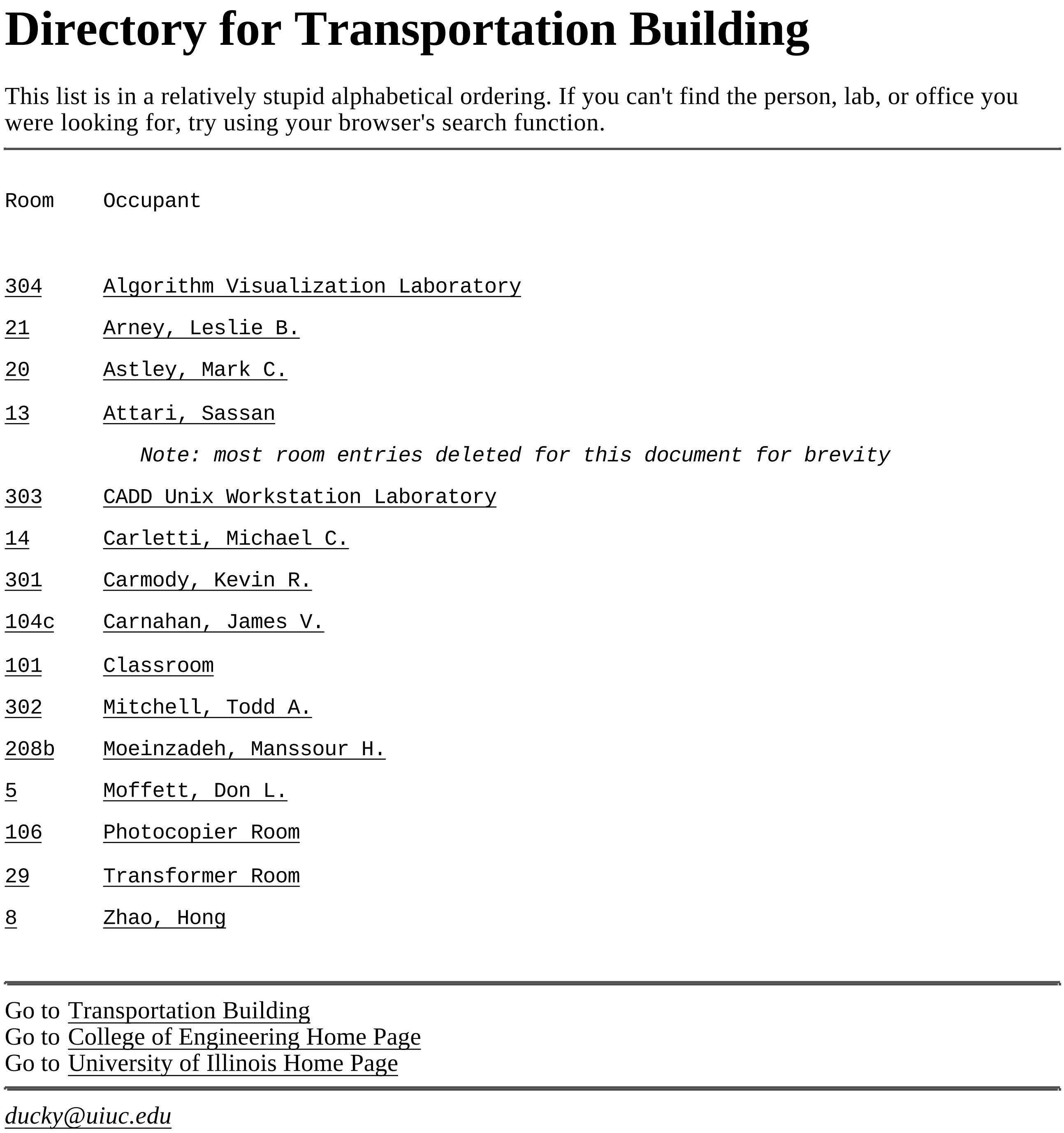
Maps and Floorplans

Eleven of the buildings that have floorplans also have occupancy information. For example, clicking on **Textual Building Directory** on the Transportation Building lobby page will take the user to a listing of all the rooms in the building, ordered by occupant, as seen in Figure 12. (Note: because the document is so long, it has been edited. Most of the rooms have been deleted.)

Furthermore, occupancy information can be determined on a room-by-room basis by clicking on the appropriate room. For example, clicking on the floorplan at room 117 (on Figure 11) will take you to the General Engineering Department's home page (Figure 13). (To get to different floors, the user can click on the stairs or the elevator.)

Once on the General Engineering Department's home page (Figure 13), clicking on the text **117 Transportation Building** will take the user to a page showing the first floor of the Transportation Building with room 117 circled (Figure 14).

This "zooming out" can continue for several more layers. Clicking on **Where am I on campus?** on the room location page (Figure 14) will take the user to a map of the Engineering campus with the Transportation Building circled (Figure 15)



Directory for Transportation Building

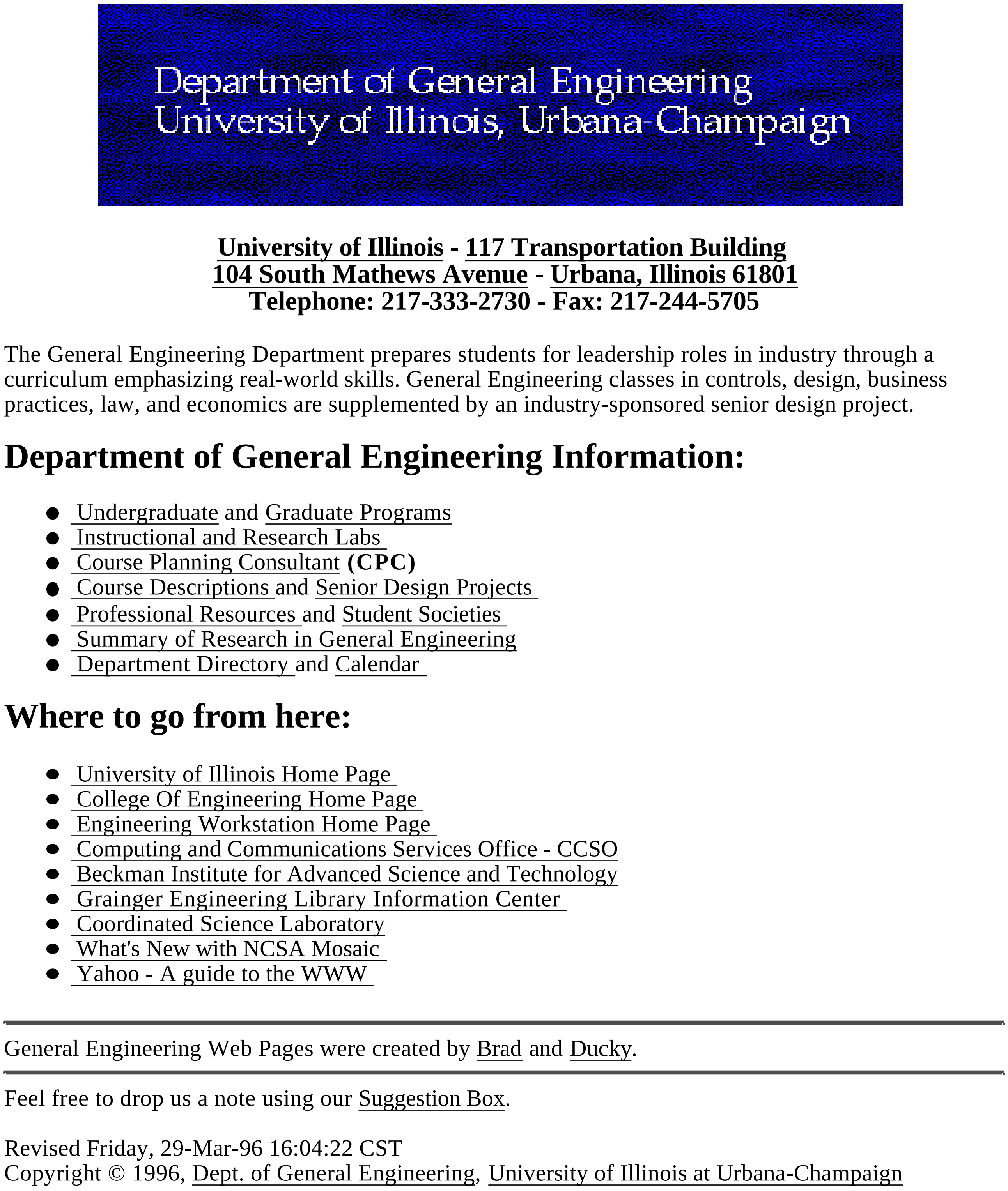
This list is in a relatively stupid alphabetical ordering. If you can't find the person, lab, or office you were looking for, try using your browser's search function.

Room	Occupant
304	Algorithm Visualization Laboratory
21	Arney, Leslie B.
20	Astley, Mark C.
13	Attari, Sassan
	Note: most room entries deleted for this document for brevity
303	CADD Unix Workstation Laboratory
14	Carletti, Michael C.
301	Carmody, Kevin R.
104c	Carnahan, James V.
101	Classroom
302	Mitchell, Todd A.
208b	Moeinzadeh, Manssour H.
5	Moffett, Don L.
106	Photocopier Room
29	Transformer Room
8	Zhao, Hong

Go to Transportation Building
Go to College of Engineering Home Page
Go to University of Illinois Home Page

ducky@uiuc.edu

Figure 12 - Building Occupancy Example



Department of General Engineering
University of Illinois, Urbana-Champaign

University of Illinois - 117 Transportation Building
104 South Mathews Avenue - Urbana, Illinois 61801
Telephone: 217-333-2730 - Fax: 217-244-5705

The General Engineering Department prepares students for leadership roles in industry through a curriculum emphasizing real-world skills. General Engineering classes in controls, design, business practices, law, and economics are supplemented by an industry-sponsored senior design project.

Department of General Engineering Information:

- Undergraduate and Graduate Programs
- Instructional and Research Labs
- Course Planning Consultant **(CPC)**
- Course Descriptions and Senior Design Projects
- Professional Resources and Student Societies
- Summary of Research in General Engineering
- Department Directory and Calendar

Where to go from here:

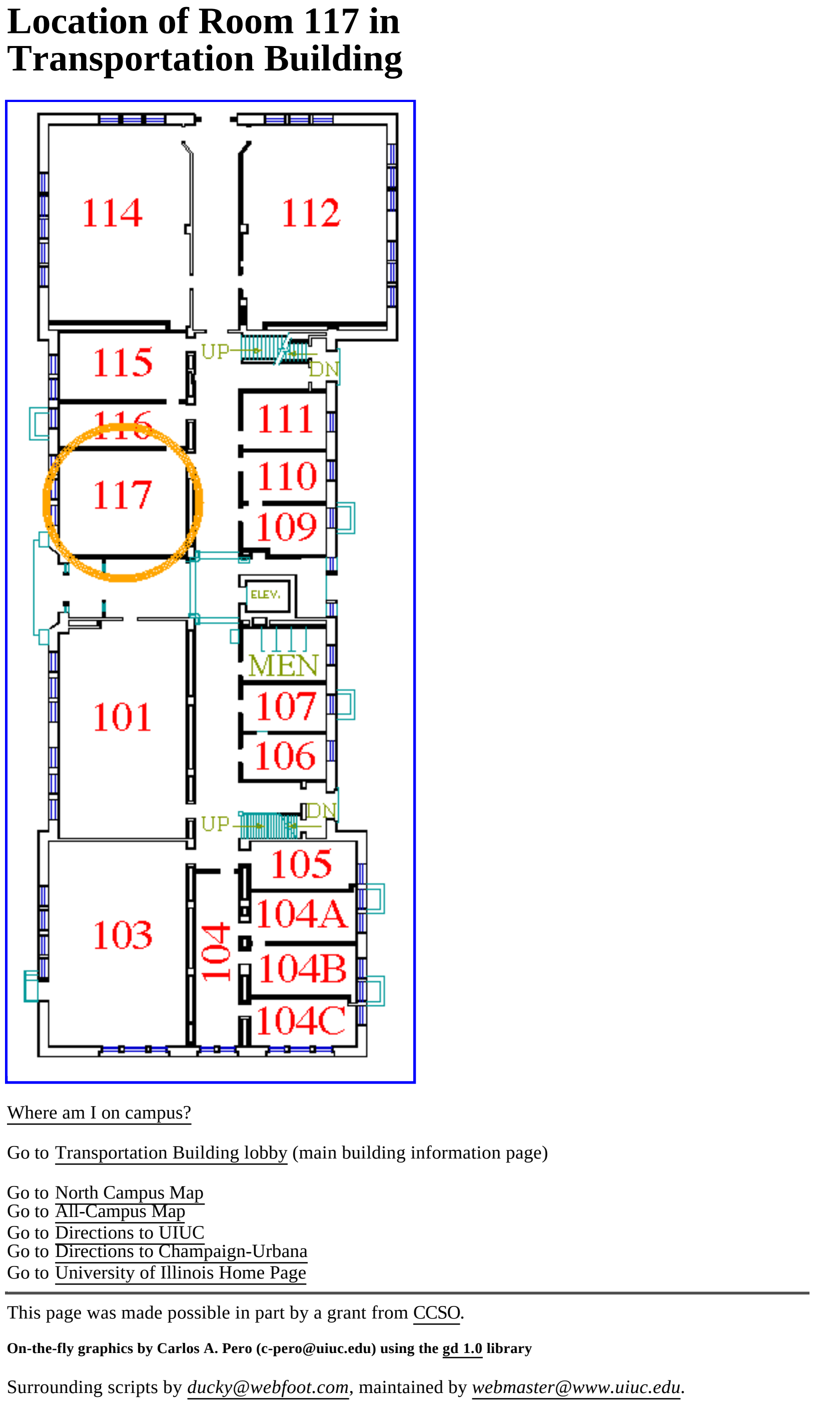
- University of Illinois Home Page
- College Of Engineering Home Page
- Engineering Workstation Home Page
- Computing and Communications Services Office - CCSO
- Beckman Institute for Advanced Science and Technology
- Grainger Engineering Library Information Center
- Coordinated Science Laboratory
- What's New with NCSA Mosaic
- Yahoo - A guide to the WWW

General Engineering Web Pages were created by Brad and Ducky.

Feel free to drop us a note using our Suggestion Box.

Revised Friday, 29-Mar-96 16:04:22 CST
Copyright © 1996, Dept. of General Engineering, University of Illinois at Urbana-Champaign

Figure 13 - General Engineering Department Home Page



Location of Room 117 in Transportation Building

Where am I on campus?

Go to Transportation Building lobby (main building information page)

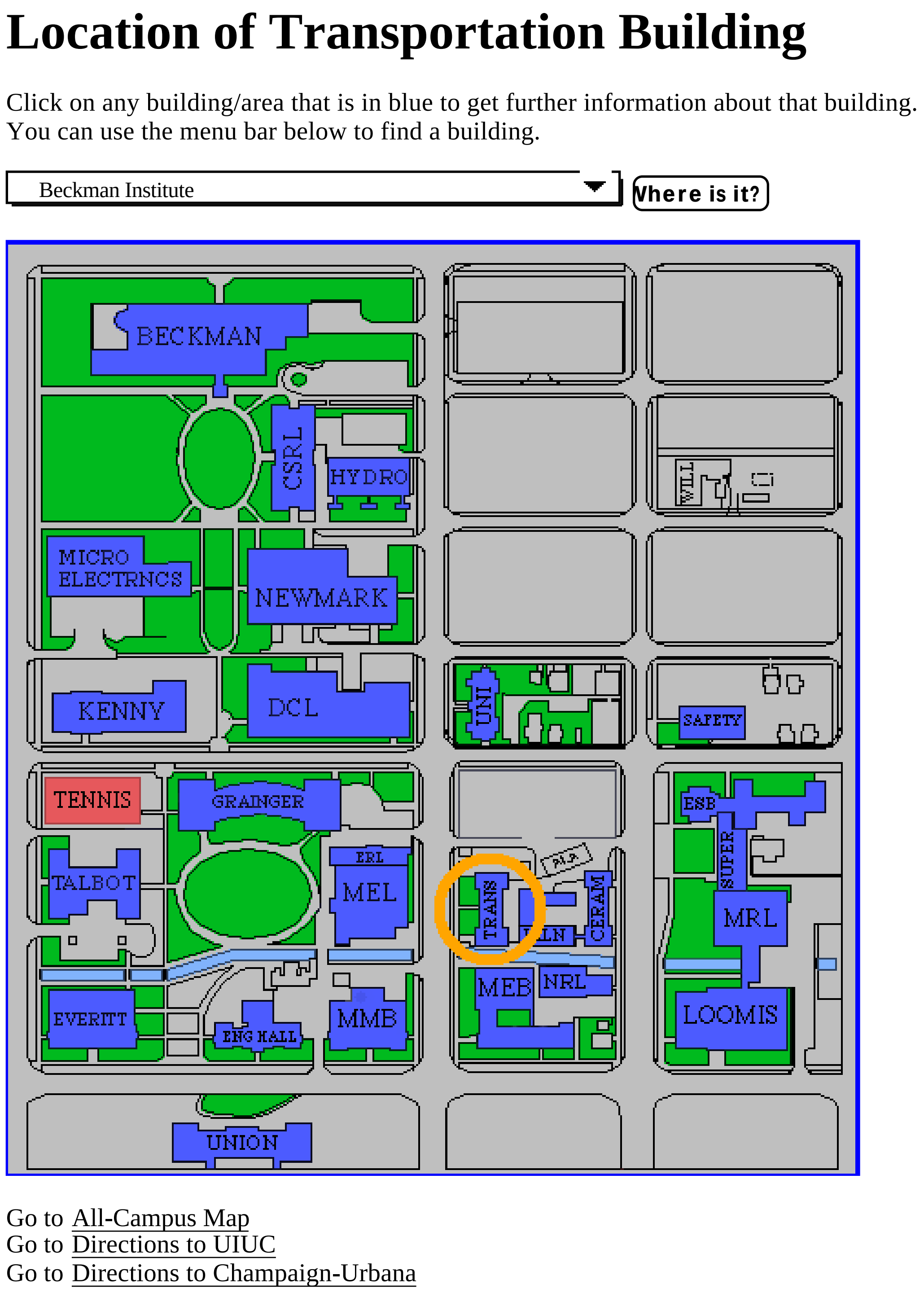
Go to North Campus Map
Go to All-Campus Map
Go to Directions to UIUC
Go to Directions to Champaign-Urbana
Go to University of Illinois Home Page

This page was made possible in part by a grant from CCSO.

On-the-fly graphics by Carlos A. Pero (c-pero@uiuc.edu) using the gd 1.0 library

Surrounding scripts by *ducky@webfoot.com*, maintained by *webmaster@www.uiuc.edu*.

Figure 14 - Room Location Example



Location of Transportation Building

Click on any building/area that is in blue to get further information about that building. You can use the menu bar below to find a building.

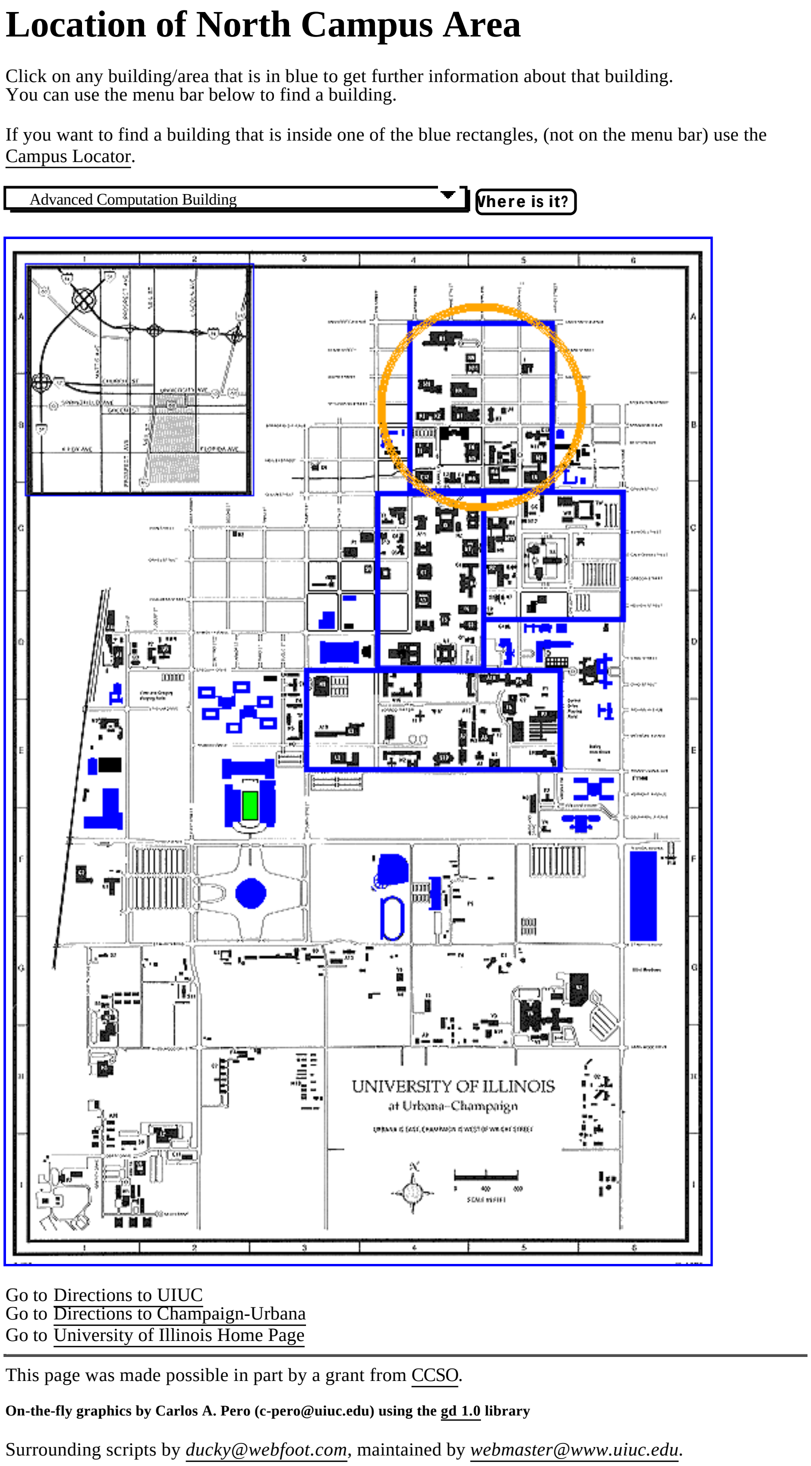
Go to All-Campus Map
Go to Directions to UIUC
Go to Directions to Champaign-Urbana

Figure 15 - Building Location Example

Clicking on **All-Campus Map** at the bottom of the building location page (Figure 15) will take the user to the all-campus map with the Engineering campus circled (Figure 16). Two more levels of "zooming out" are possible to show the location of the campus in relation to Champaign and Urbana, and the location of Champaign-Urbana in Champaign County.

"Zooming in" is also possible. From the all-campus map (Figure 16), clicking on the top rectangle will take the user to the Engineering campus map (Figure 15 but without the circle). From the Engineering campus map, clicking on the Transportation Building will take the user to the Transportation Building lobby (Figures 11).

All 134 buildings with information can be "zoomed in" on in this manner, and all the buildings' locations can be found by "zooming out" from their respective lobbies, with circles surrounding the building of interest the appropriate map - one of four area maps, or on the all-campus map for those outside the main campus core.



Location of North Campus Area

Click on any building/area that is in blue to get further information about that building.
You can use the menu bar below to find a building.

If you want to find a building that is inside one of the blue rectangles, (not on the menu bar) use the Campus Locator.

Go to Directions to UIUC
Go to Directions to Champaign-Urbana
Go to University of Illinois Home Page

This page was made possible in part by a grant from CCSO.

On-the-fly graphics by Carlos A. Pero (c-pero@uiuc.edu) using the gd 1.0 library

Surrounding scripts by *ducky@webfoot.com*, maintained by *webmaster@www.uiuc.edu*.

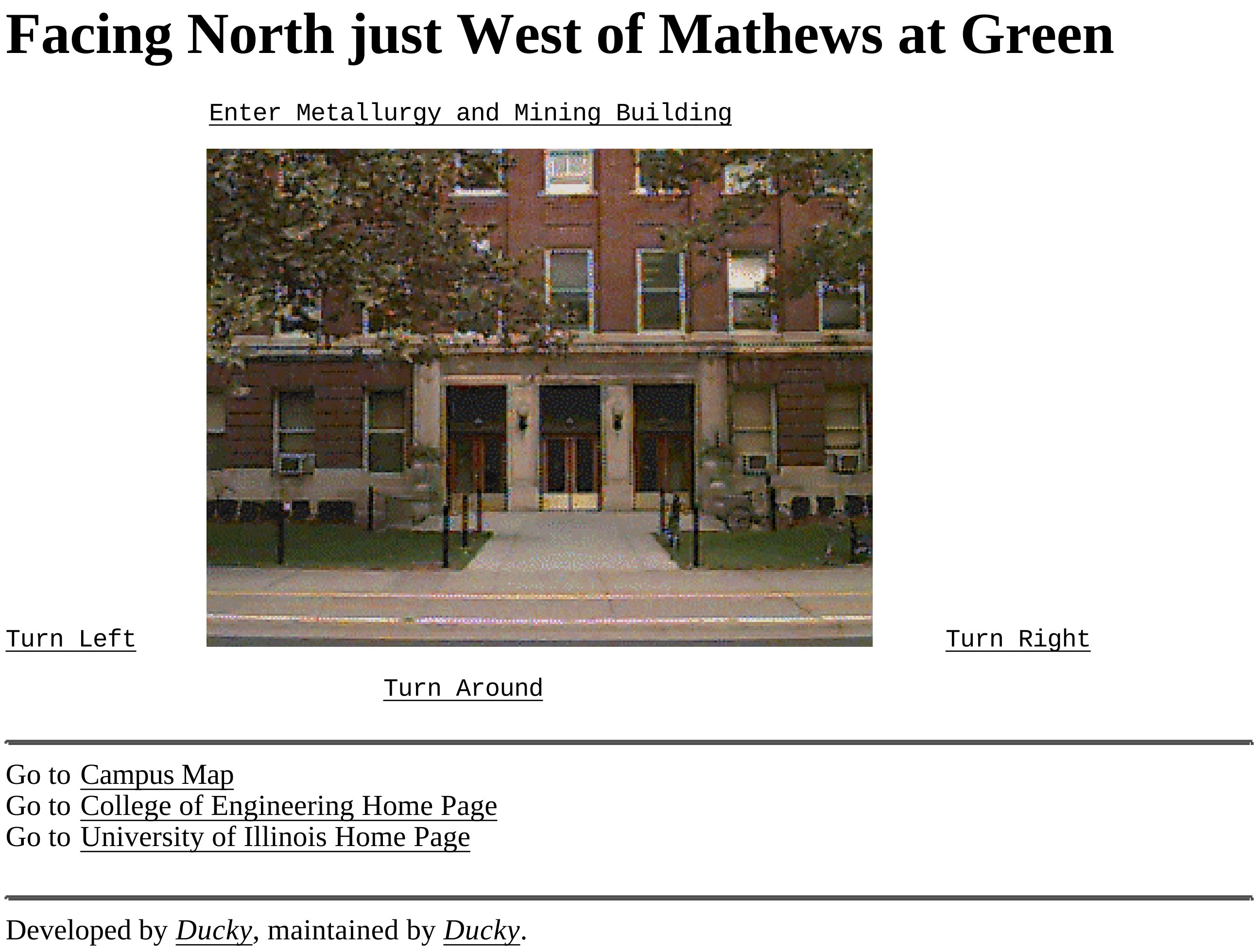
Figure 16 - Area Map Location Example

The URLs for these maps and floorplans were deliberately kept simple. It would have been relatively simple to write code that took the name of a GIF file and x-y coordinates from a URL and based all manipulations on that information. However, human beings don't think naturally in terms of pixels on an image. Humans think in terms of rooms numbers and buildings. Thus, extra work was done so that the information embedded in the URL would be the building and the room number instead of the location of the GIF file and the bounding box of that room.

Care was also taken to allow for appropriate fallbacks. If the location of a room in a building is requested, but no floorplans is ready for that building, the program will return instead the location of the building. This is done at the time of the access, so that people can make links to rooms that don't have floorplans. Once the floorplans become available for that building, the person's old link will suddenly show the room circled instead of the building. No intervention on the author's part is required.

Virtual Walkthrough

A "virtual walkthrough" of the campus is available, allowing people to get a feel for the look of the campus without needing to visit in person. A photo of a scene is displayed with navigational directions surrounding the picture. For example, Figure 17 shows what someone would see if they were facing north just west of Mathews Avenue at Green Street.



Facing North just West of Mathews at Green

Enter Metallurgy and Mining Building

Turn Left

Turn Right

Turn Around

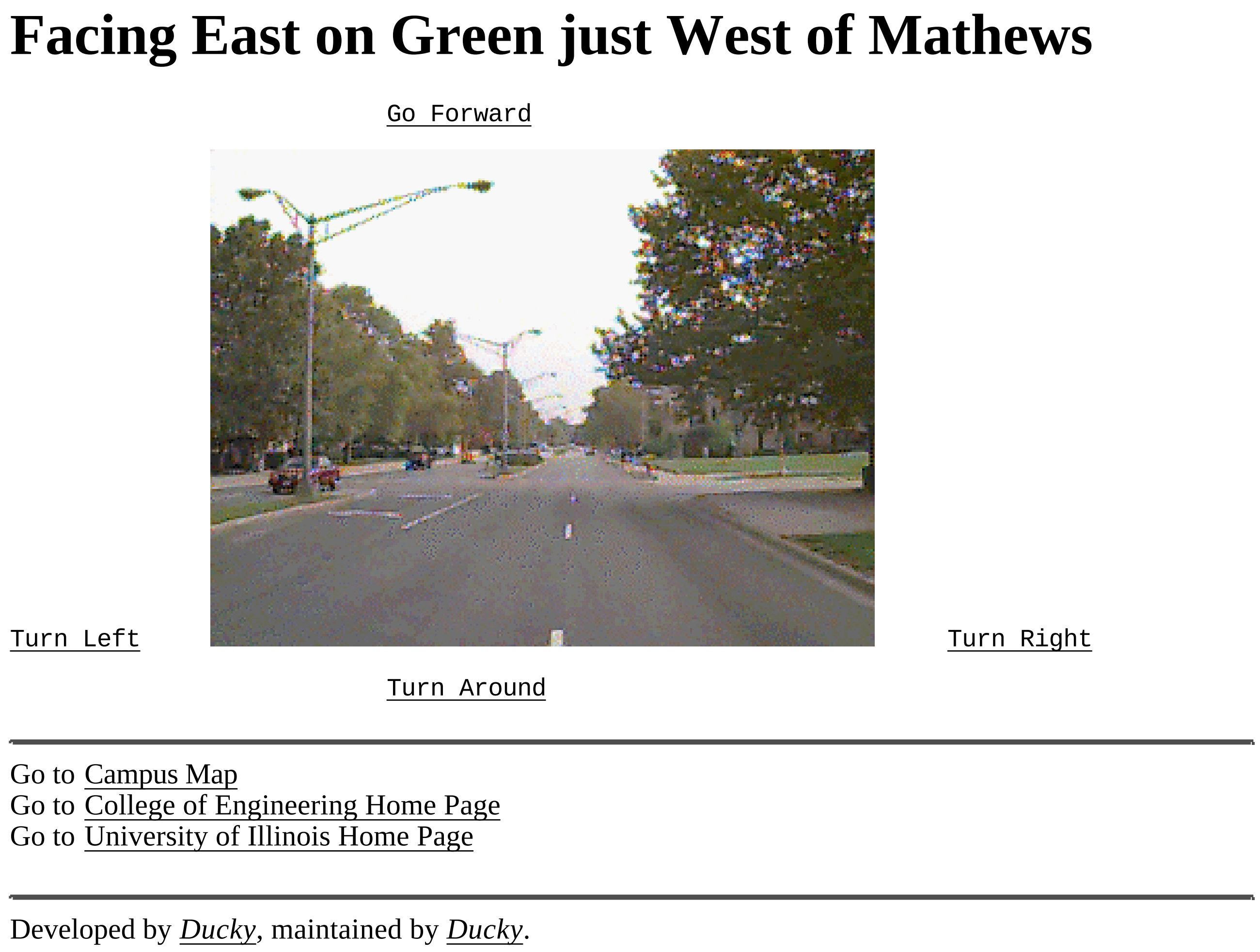
Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

Developed by *Ducky*, maintained by *Ducky*.

Figure 17 - Virtual Walkthrough Example

Clicking on **Enter Metallurgy and Mining Building**, will take the user to the lobby for the Metallurgy and Mining Building.

Clicking on **Turn Right** will take the user to a photo showing what the user would see if they turned right from their virtual standpoint, namely a photo facing east on Green Street just west of Mathews Avenue (Figure 18).



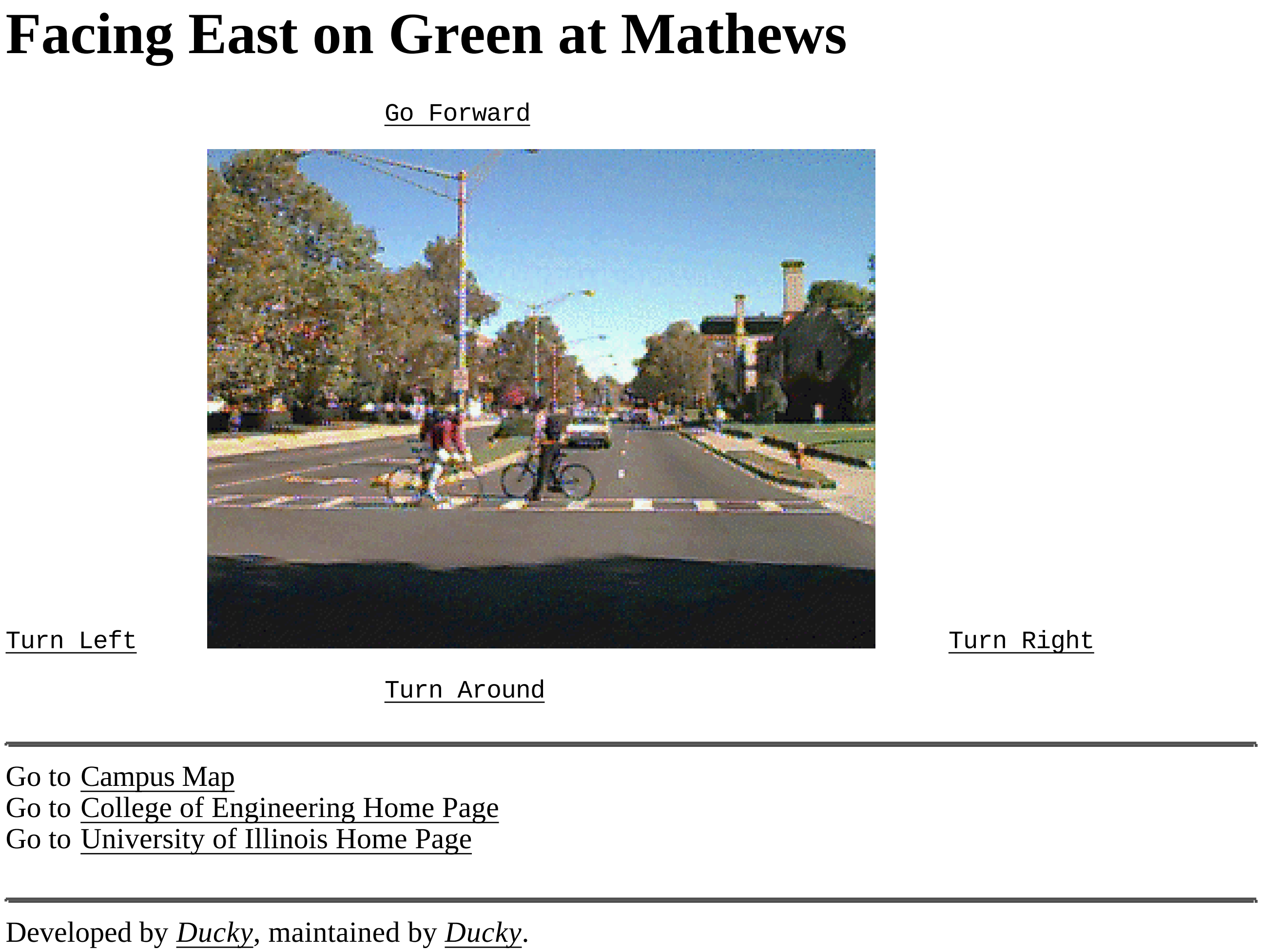
Facing East on Green just West of Mathews

Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

Developed by *Ducky*, maintained by *Ducky*.

Figure 18 - Virtual Walkthrough Example

Clicking on **Go Forward** will take the user to a photo facing east on Green at Mathews (Figure 19), and so on.



Facing East on Green at Mathews

Go Forward

Turn Left

Turn Right

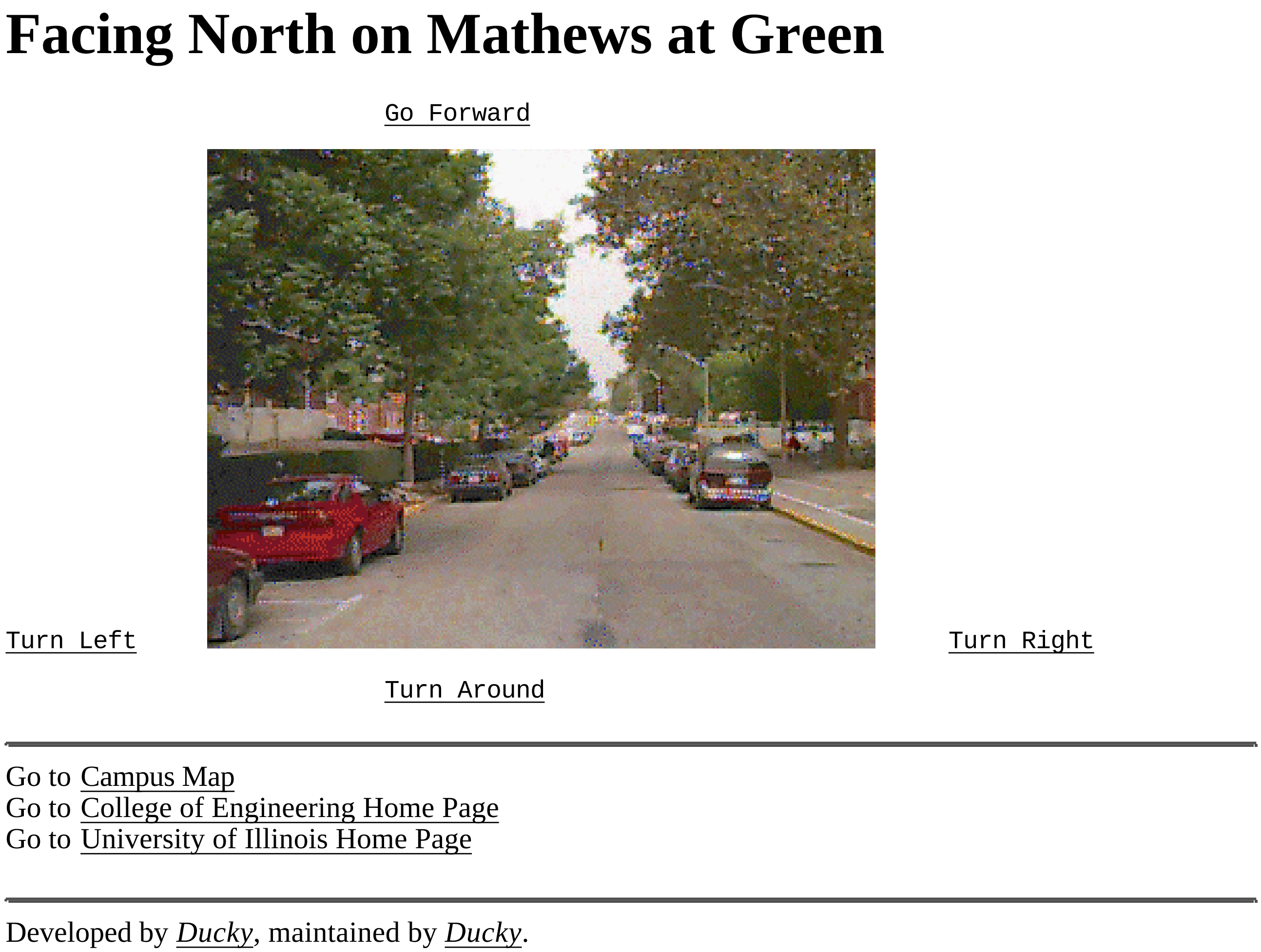
Turn Around

Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

Developed by *Ducky*, maintained by *Ducky*.

Figure 19 - Virtual Walkthrough Example

The sequence from Figure 20 to Figure 24 shows a continuation of this virtual stroll to the front of the Transportation Building.



Facing North on Mathews at Green

Go Forward

Turn Left

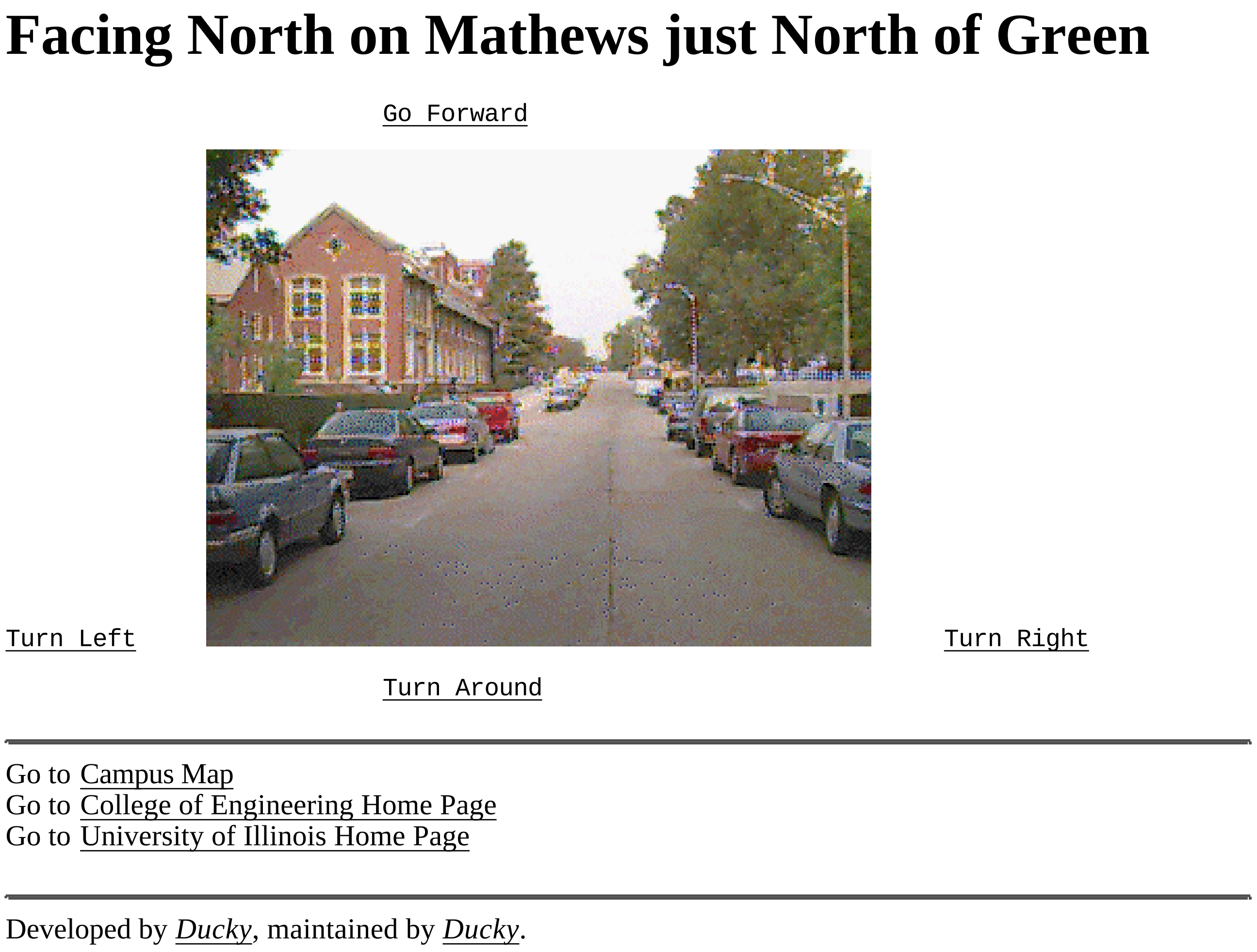
Turn Right

Turn Around

Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

Developed by *Ducky*, maintained by *Ducky*.

Figure 20 - Virtual Walkthrough Example



Facing North on Mathews just North of Green

Go Forward

Turn Left

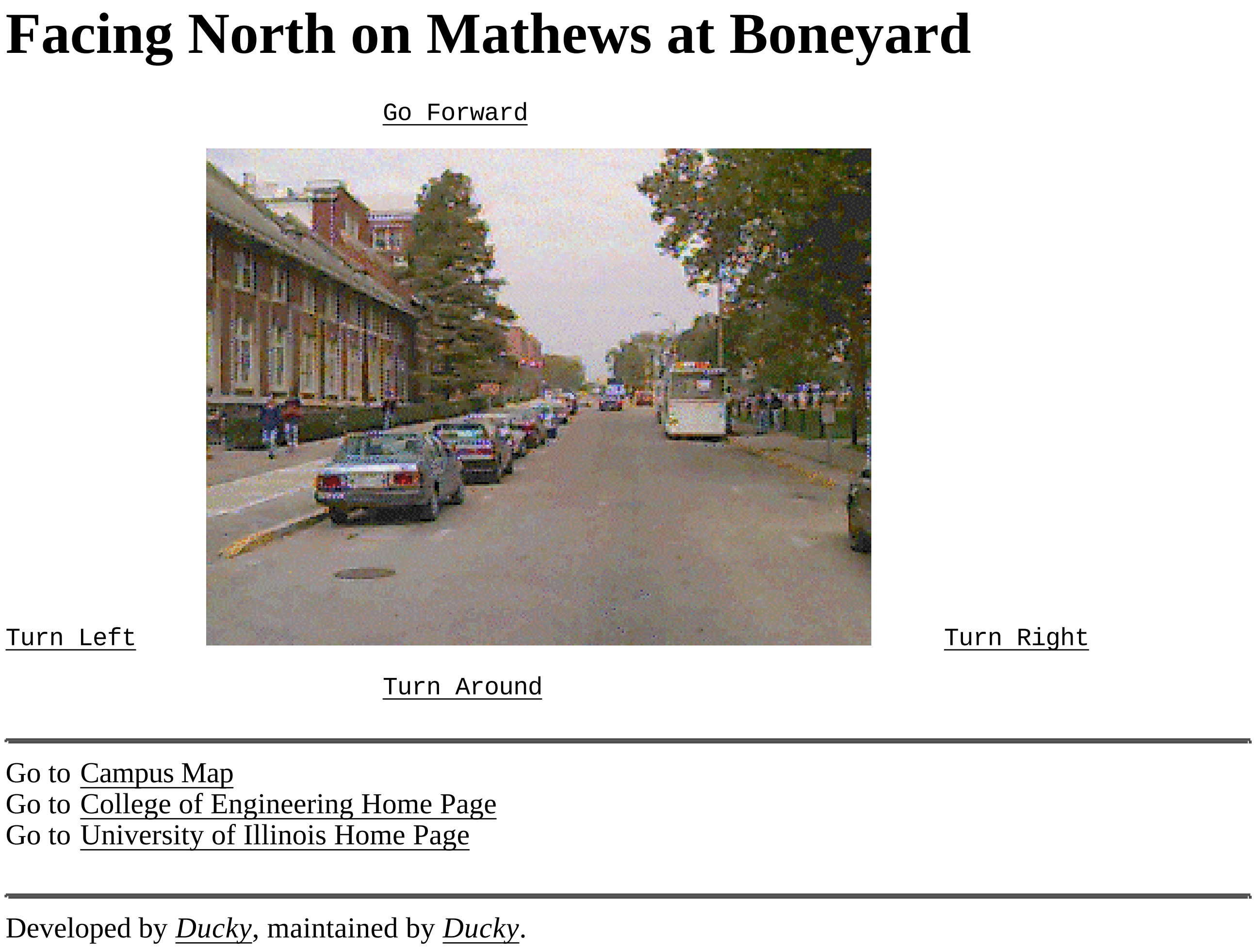
Turn Right

Turn Around

Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

Developed by *Ducky*, maintained by *Ducky*.

Figure 21 - Virtual Walkthrough Example



Facing North on Mathews at Boneyard

Go Forward

Turn Left

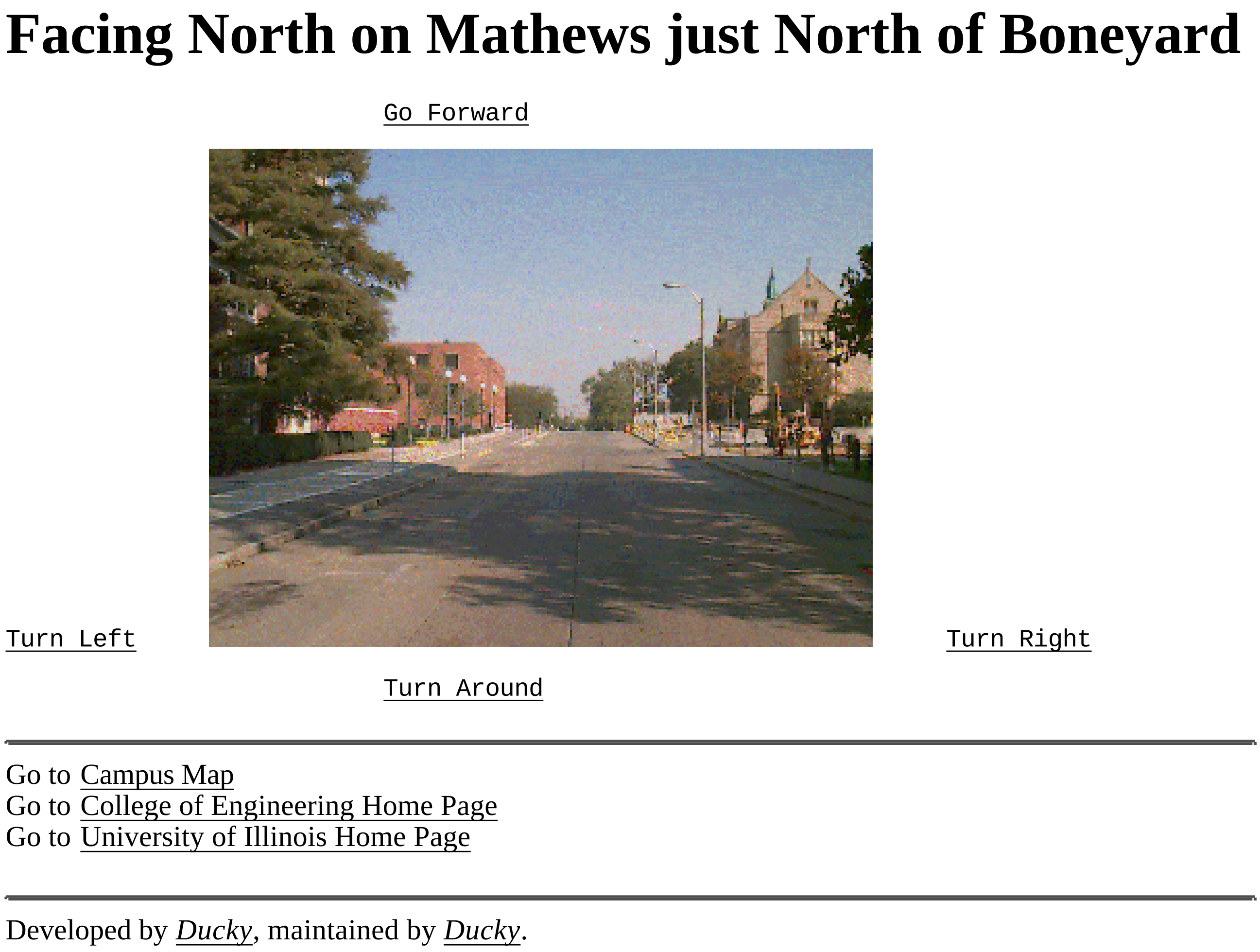
Turn Right

Turn Around

Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

Developed by *Ducky*, maintained by *Ducky*.

Figure 22 - Virtual Walkthrough Example



Facing North on Mathews just North of Boneyard

Go Forward

Turn Left

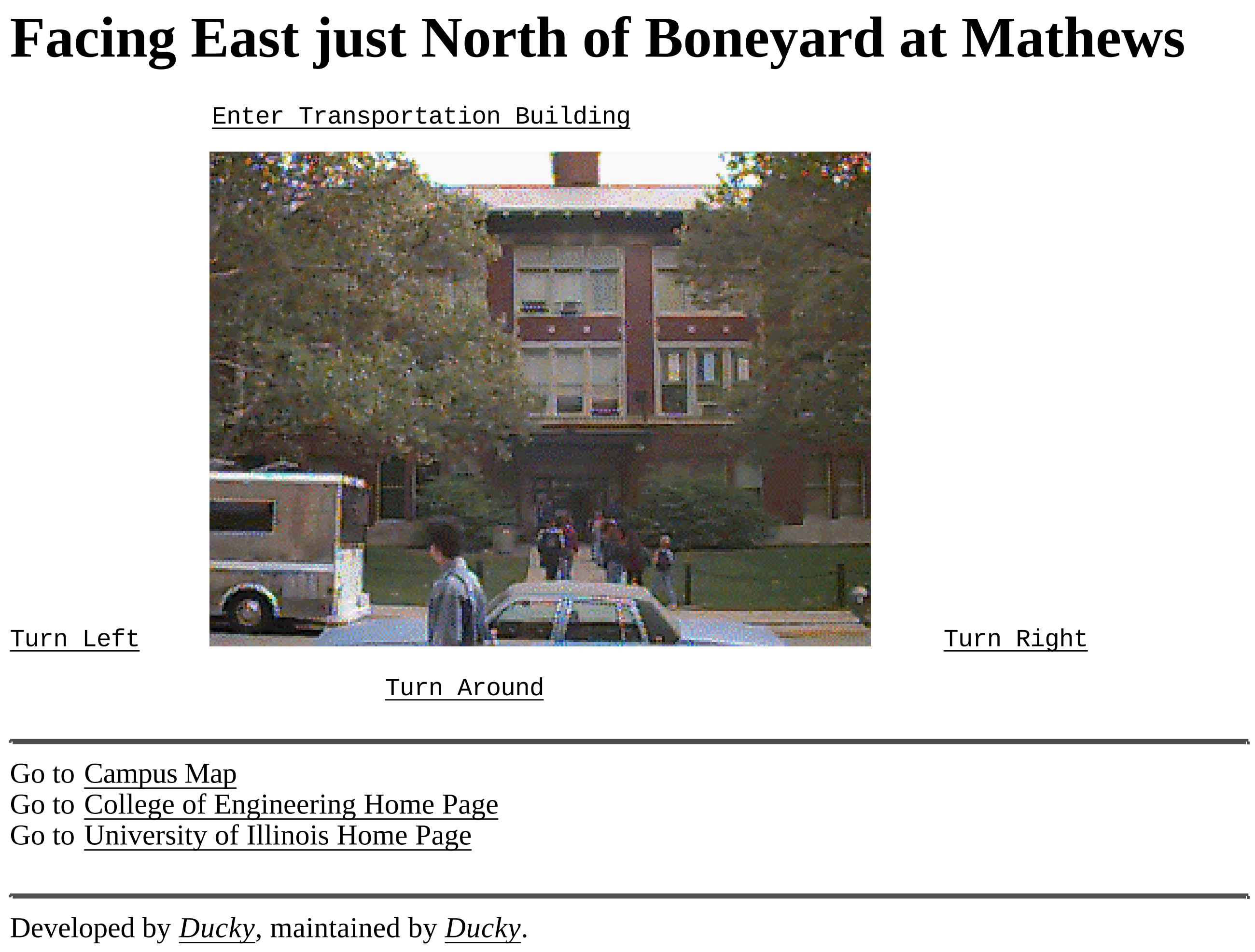
Turn Right

Turn Around

Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

Developed by *Ducky*, maintained by *Ducky*.

Figure 23 - Virtual Walkthrough Example



Facing East just North of Boneyard at Mathews

Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

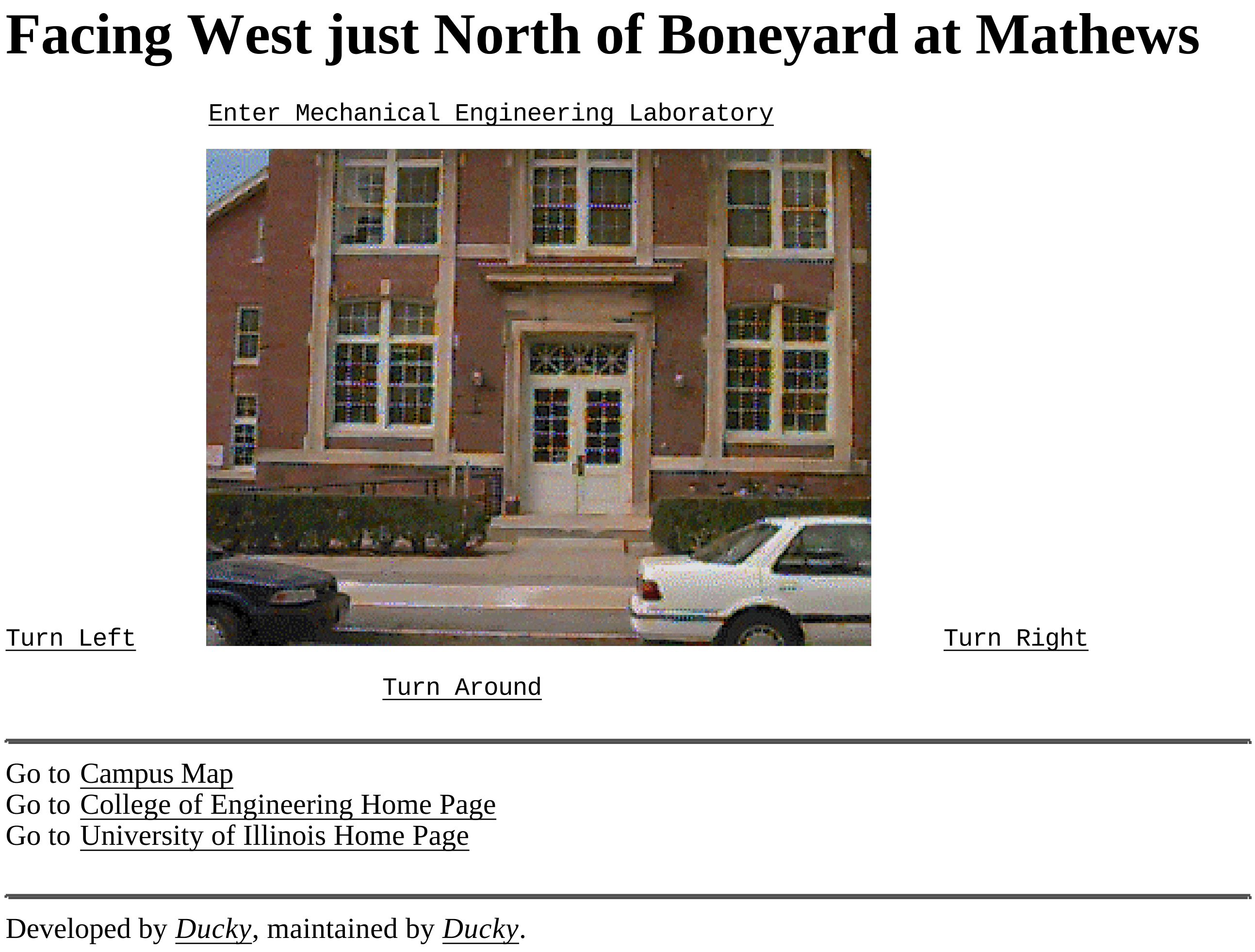
Developed by *Ducky*, maintained by *Ducky*.

Figure 24 - Virtual Walkthrough Example

Clicking on **Enter Transportation Building** from Figure 24 will take the user again to the Transportation Building lobby (Figure 11).

In all, there are 800 pages in the virtual tour - four compass directions at 200 geographical points. The points are, for the most part, a half a block apart, and cover the University property between University Avenue on the north, Sixth Street on the west, Taft on the south, and Goodwin on the east.

The tour can be accessed in several ways. Aside from typing in a URL or selecting a link from the navigation index page, clicking near the entrance of a building on a floorplan can take the user into a tour. For example, clicking near the west door of the Transportation Building floorplan in Figure 11 will take the user to a view from the west door (Figure 25).



Facing West just North of Boneyard at Mathews

Enter Mechanical Engineering Laboratory

Turn Left

Turn Right

Turn Around

Go to Campus Map
Go to College of Engineering Home Page
Go to University of Illinois Home Page

Developed by *Ducky*, maintained by *Ducky*.

Figure 25 - Virtual Walkthrough Example

(This is happens to be the same page that the user would come to by clicking on **Turn Around** in Figure 24.) Users can also click on a street in an area map, and for most locations on the Quad, Engineering campus, and Agriculture/Business maps, will be taken to the appropriate point in the virtual walkthrough.

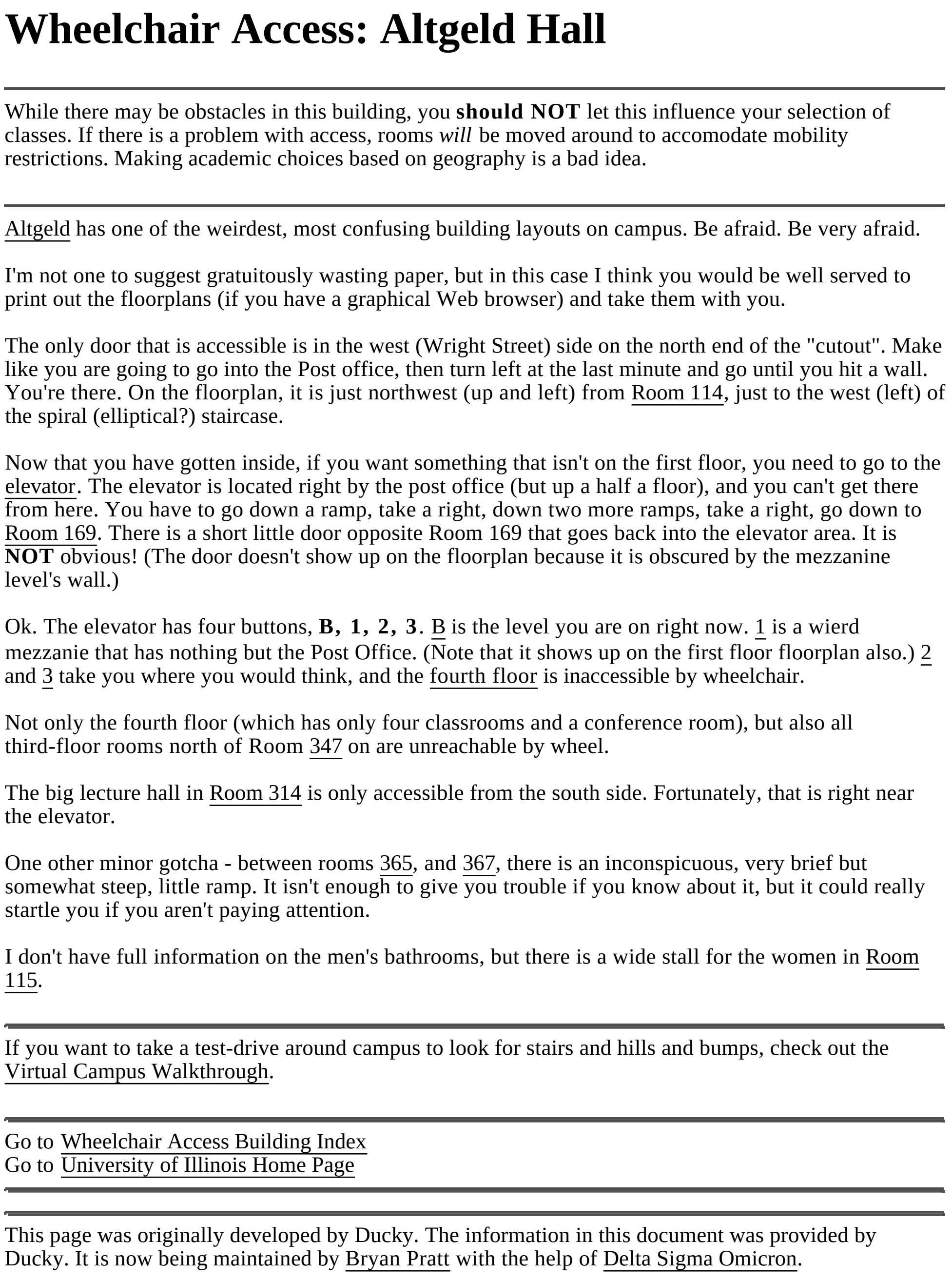
Wheelchair Access

The wheelchair access pages, available from many of the building lobbies, list elevators, stairs, ramps, automatic door openers, bathrooms, and any other obstacles that people with mobility restrictions could encounter. Furthermore, because rooms could be circled via the above system, links to exact locations could be made in the text of the access page. (See Figure 26 for an example of access information for Altgeld Hall, a particularly confusing building.)

The wheelchair pages were definitely an afterthought, something that turned out to be easy to do once all the other pieces were developed. They ended up yielding the most emotional appreciation and an award from the disabled students organization. The wheelchair access information also has a much broader utility than I had originally thought: information about ramps and elevators is very useful when moving equipment between buildings, or even for freshmen trying to figure out the vagaries of old, confusing buildings.

Navigational System Features Summary

The navigational system allows users to navigate easily through a wide array of information. Information on how to get to a room, how to get to a building, what will be seen on the trip, and what one will find once there is seamlessly integrated.



Wheelchair Access: Altgeld Hall

While there may be obstacles in this building, you **should NOT** let this influence your selection of classes. If there is a problem with access, rooms *will* be moved around to accomodate mobility restrictions. Making academic choices based on geography is a bad idea.

Altgeld has one of the weirdest, most confusing building layouts on campus. Be afraid. Be very afraid.

I'm not one to suggest gratuitously wasting paper, but in this case I think you would be well served to print out the floorplans (if you have a graphical Web browser) and take them with you.

The only door that is accessible is in the west (Wright Street) side on the north end of the "cutout". Make like you are going to go into the Post office, then turn left at the last minute and go until you hit a wall. You're there. On the floorplan, it is just northwest (up and left) from Room 114, just to the west (left) of the spiral (elliptical?) staircase.

Now that you have gotten inside, if you want something that isn't on the first floor, you need to go to the elevator. The elevator is located right by the post office (but up a half a floor), and you can't get there from here. You have to go down a ramp, take a right, down two more ramps, take a right, go down to Room 169. There is a short little door opposite Room 169 that goes back into the elevator area. It is **NOT** obvious! (The door doesn't show up on the floorplan because it is obscured by the mezzanine level's wall.)

Ok. The elevator has four buttons, **B, 1, 2, 3**. B is the level you are on right now. 1 is a wierd mezzanie that has nothing but the Post Office. (Note that it shows up on the first floor floorplan also.) 2 and 3 take you where you would think, and the fourth floor is inaccessible by wheelchair.

Not only the fourth floor (which has only four classrooms and a conference room), but also all third-floor rooms north of Room 347 on are unreachable by wheel.

The big lecture hall in Room 314 is only accessible from the south side. Fortunately, that is right near the elevator.

One other minor gotcha - between rooms 365, and 367, there is an inconspicuous, very brief but somewhat steep, little ramp. It isn't enough to give you trouble if you know about it, but it could really startle you if you aren't paying attention.

I don't have full information on the men's bathrooms, but there is a wide stall for the women in Room 115.

If you want to take a test-drive around campus to look for stairs and hills and bumps, check out the Virtual Campus Walkthrough.

Go to Wheelchair Access Building Index
Go to University of Illinois Home Page

This page was originally developed by Ducky. The information in this document was provided by Ducky. It is now being maintained by Bryan Pratt with the help of Delta Sigma Omicron.

Figure 26 - Wheelchair Access Example

7.2 - NAVIGATION SYSTEM IMPLEMENTATION

The navigation system, with its maps, floorplans, wheelchair access information, and virtual tour, required a substantial amount of code development, image translation, and information gathering. This section explains how the major subsystems of the navigational aids were implemented, broken out by subsystem.

Maps and Floorplans

The most complicated subsystem was the maps and floorplans segment. Not only did maps and floorplans need to be found, but some translation had to be done to make them useful. Code then had to be developed to do inward and outward zooming, and occupancy data ferreted out.

Acquisition of Information

The length of time it took to find the source of floorplans was perhaps indicative of how inaccessible good navigational information is. It took four months of casual but steady investigation to find a set of floorplans at the Operations and Maintenance `ftp` site.

Occupancy information (needed for zooming in to rooms) was decentralized to the point of being inaccessible: I resorted to departmental phone lists and ambulatory inspections of the buildings.

The all-campus map was scanned in from a map developed by the Office of Facilities Planning and Management. The Engineering campus map was captured from a map developed by the College of Engineering's Office for Financial Programs. The other area maps were derived from blow-ups of the all-campus map.

Image Translation

Some manipulation was required for the maps, which were all originally in black and white. The buildings that had lobbies were colored blue, and the regions that had area maps were given blue boundaries on the all-campus map. Touchups were done on all the area maps, with labeling or relabeling of most buildings, coloring streets and sidewalks grey, and coloring grassy areas green. Various paint programs, such as `xpaint` and `Photoshop` were used for the map manipulations.

It was necessary to do quite a bit more translation to obtain useful floorplans.

- The original drawings were in `AutoCAD` format, which is not recognized by any browsers. They had to be converted to GIF format.
- The original drawings frequently did not have north facing the top of the page. Because the floorplans were tightly integrated with maps that show north facing up, leaving the floorplans at an arbitrary orientation would have been confusing.
- The room numbers were invariably too small to be legible when the floorplan was a size that was reasonable for a browser to display.
- The `AutoCAD` drawings were red on a black background. This looks good on a computer monitor, but does not print well.

`AutoCAD` was used to remove the drawing layer corresponding to the room numbers, resize, and rotate the drawing. The shareware program `xpaint` by David

Koblas was used to put larger room numbers on the drawings. The color manipulation facilities of the shareware program `xv` by John Bradley were used to convert the image format and to change the colors of the drawing.

To ensure the accuracy of the floorplans, each building was subjected to an on-site inspection. While the accuracy of the floorplan was verified, the locations of bathrooms, department offices, conference rooms, classrooms, elevators, mechanical closets, janitorial closets, photocopiers, computer laboratories, and many other features were also noted.

Programming Implementation

The maps/floorplans system is glued together by two perl scripts: `who_is_in` and `where_is`. The former script takes the user to the home page(s) of the occupant(s) of the room or building requested. The latter script creates an HTML page and GIF-format image that show the user the location of the room or building requested. In other words, `where_is` is for zooming out; `who_is_in` is for zooming in to rooms.

who_is_in

There is a standard way of making "clickable maps" on the Web: images, that when clicked on in different coordinates, will take you to different pages. This "ismap" format is

```
URL    x1,y1   x2,y2
```

where `URL` is the URL of the page that should be returned if the user selects a point inside the box bounded by coordinates `x1,y1` and `x2,y2`. It would clearly be convenient for the user to be able to click on the image of a floorplan and be given the home page of that room's occupant. It would also clearly be useful to have the room number associated with the coordinates, for use by `where_is`. The script `who_is_in` was developed to allow the room number to be attached to the coordinates while still returning the proper home page(s). A level of indirection is employed; instead of the map data file containing lines like

```
http://www.webfoot.com/ducky.home.html  123,456 321,654
```

the file instead includes lines like

```
http://www.uiuc.edu/cgi-bin/who_is_inin?bldg=grainger&room=451    123,456 321,654
```

A separate data file for each building connects the room number to the occupant's home page with additional information:

```
ccso    451     http://www.webfoot.com/ducky.home.html  Sherwood, Ducky
```

Note that the image coordinates corresponding to the location of the rooms' walls, need only change when buildings are remodeled. This is usually an infrequent event. The occupancy information, which changes frequently, is arranged by room number. Segregating the rarely changing information from the frequently changing information simplifies maintenance; making the frequently changed information in well-understood terms eases maintenance even more.

The first field gives the "owner" of the room, so that buildings that are shared by more than one academic/administrative unit can be more easily updated. The second field gives the room number. The third field gives the URL of the occupant, and everything past that (a variable-length field) is the name of the occupant.

The script `who_is_in` determines which data file to use (based on the building name), then searches through it for the appropriate room number. Once that is found, it pulls out the URL and occupant. If there is more than one match for the room number, i.e. there is more than one occupant of a room, then an HTML page is generated with a list of all the occupants, with hyperlinks to each person's home page. If there is only one occupant, then the user is given that occupant's page directly.

If no room is specified, `who_is_in` notes all the lines and returns a listing of all the occupants in the building, as shown in Figure 12. The perl code for `who_is_in` is given in Appendix A.

where_is

The `where_is` script creates a page by looking in the ismap file for the coordinates of the room. This can be determined by doing a text search for the building and room, e.g. `bldg=grainger&room=451`, which would find the following line.

```
http://www.uiuc.edu/cgi-bin/who_is_in?bldg=grainger&room=451    123,456 321,654
```

From the coordinates of the box bounding the active region, the center point of the room and the radius of a circle containing the room or building can be determined.

That information is passed to another script, `where_is.gd` written by Carlos Pero, which creates a temporary GIF of the relevant image with the appropriate room circled, as in Figure 14.

Note that care must be taken when dealing with temporary GIF files. First, because most browsers cache images, the name must be distinct enough that users are not likely to encounter a second image with the same name before the cache is cleared out. Second, the temporary image must be deleted at some point, but not before the completion of the script. Because most browsers do not download images until the text of a page has been fully downloaded, removing the image too soon will result in the browser not finding the image at all.

If the pixel-to-room data for the requested floor does not exist, or if a room is not specified, the code returns an image of the building circled on the appropriate map, as in Figure 15. Thus authors can link to a room in a building, even if the floorplan information does not yet exist, and give visitors to their pages useful information. When the floorplan becomes available, that hyperlink will switch from showing the building circled to showing the room circled.

Wheelchair Access Information

As an inspection of the buildings was done to check the floorplans for accuracy and to note locations of important rooms, it was a simple matter to also note the locations of wheelchair bathrooms, elevators, chairlifts, ramps, and automatic doors. This information was used to develop access information pages (see Figure 26). Additional information was gleaned from interviews with students with mobility restrictions.

In addition to a textual description of the buildings' obstacles, links to the floorplans were used when possible. Thus, the precise location of e.g. elevators could be shown on a floorplan.

Virtual Tour

A digital camera was used to take the photos for the virtual tour because of the speed of acquisition and low cost. Photographs were taken in all four compass directions at half-block intervals. Because the University of Illinois is located on topographically uninteresting terrain, the streets are in a near-perfect grid. By imparting some information about the order of the streets to a script, the locations of buildings, and the relationship of forward, left, right, backwards, north, south, east, and west to each other, it was relatively straightforward to write code that generated hypertext to frame the photos.

7.3 - Navigational System Summary

The navigational system provides a very rich set of resources for finding one’s way around campus. Buildings can be located on maps. Rooms can be located in buildings. Even the stroll from one building to another can be simulated.   There is also information about the buildings to act as a bridge between different navigational elements. Furthermore, the implementation was such that it is very easy to make links to these maps and floorplans, so they can be incorporated into other resources. Because of that, extending the system to include wheelchair access information was possible.

8.0 - STUDENT REGISTRATION MATERIALS

The Courses Catalog, Timetable, and Programs of Study are the three primary documents that students use while on-campus. The Programs of Study contains graduation requirements, including lists of applicable courses. The Courses Catalog contains descriptions of courses available. The Timetable contains a list of classes available in a given term, including the time, location, and (when available) the name of the instructor. Thus, every semester, students must refer simultaneously to all three documents. It was felt that a hypertextually linked version of the three would be superior to the three paper documents separately. This section will show the features and implementation of the UIUC on-line registration materials.

8.1 - Registration Materials Features

The *Programs of Study* contains a wealth of information about the university, including the graduation requirements. The amount of flexibility a student has varies widely from major to major, but usually is presented in a tabular form. The electronic form of the *Programs of Study* contains all of the information in the printed version. An example of the first-year requirements for the Curriculum in General Engineering is shown in Figure 27.

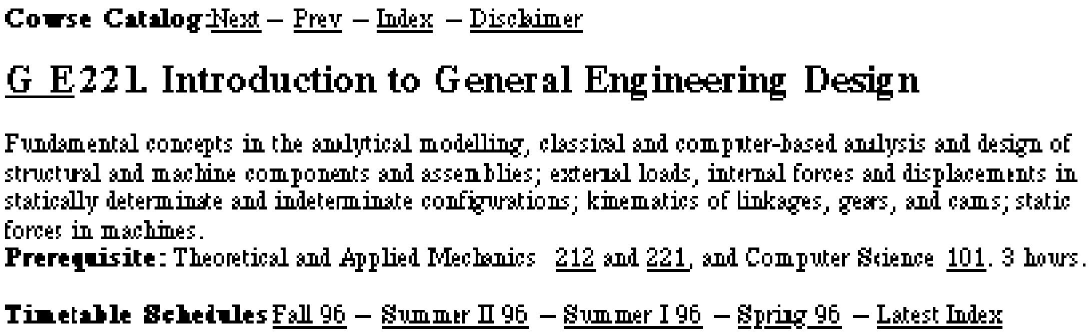
First year

HOURS	**FIRST SEMESTER** ***see footnote 1***
4	C HEM 101--General Chemistry
3	E CON 102--Microeconomic Principles, or E CON 103--Macroeconomic Principles (General education elective *see footnote 2*)
0	EN G 100--Engineering Lecture
3	G E 103--Engineering Graphics and Design
5	MATH 120--Calculus and Analytic Geometry, I
15	Total
HOURS	**SECOND SEMESTER**
3	MATH 130--Calculus and Analytic Geometry, II
2	MATH 225--Introductory Matrix Theory
4	PHYCS 106--General Physics (Mechanics)
4	R HET 105--Principles of Composition
3	Elective in social sciences or humanities *see footnote 2*
16	Total

Figure 27 - First-Year General Engineering Requirements

All of the course names in the *Programs of Study* are linked to further information about the course. For example, clicking on **GE 221** takes the user to the course description, as seen in Figure 28. In addition to the brief summary of the course's content and hyperlinked prerequisites, there are links to schedule information.

Clicking on **Fall '96** on the course description page, for example, gives the user a page with a list of all of the available sections, with the times, locations, and teachers of those sections, as seen in Figure 29. In addition to links back to the course description and to prerequisite classes, the class location and instructor are both linked. Clicking on the instructor pulls information about the instructor from the phonebook (`ph`) database, formats it nicely, and returns that information to the user. (Note that the `ph` database software predates my time at UIUC. Its discussion is outside the scope of this document.)



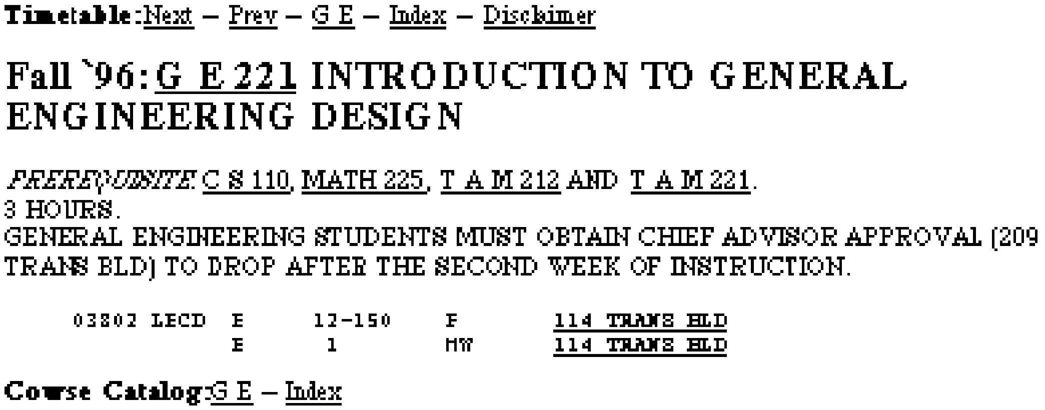
Course Catalog: Next – Prev – Index – Disclaimer

G E 221 Introduction to General Engineering Design

Fundamental concepts in the analytical modelling, classical and computer-based analysis and design of structural and machine components and assemblies; external loads, internal forces and displacements in statically determinate and indeterminate configurations; kinematics of linkages, gears, and cams; static forces in machines.
Prerequisite: Theoretical and Applied Mechanics 212 and 221, and Computer Science 101. 3 hours.

Timetable Schedules Fall 96 – Summer II 96 – Summer I 96 – Spring 96 – Latest Index

Figure 28 - Course Description Example



Timetable: Next – Prev – G E – Index – Disclaimer

Fall `96: G E 221 INTRODUCTION TO GENERAL ENGINEERING DESIGN

PREREQUISITE: C S 110, MATH 225, T A M 212 AND T A M 221.
3 HOURS.
GENERAL ENGINEERING STUDENTS MUST OBTAIN CHIEF ADVISOR APPROVAL (209 TRANS BLD) TO DROP AFTER THE SECOND WEEK OF INSTRUCTION.

```
    03802 LECD  E     12-150    F       114 TRANS BLD
                E      1        MW      114 TRANS BLD
```

Course Catalog: G E – Index

Figure 29 - Timetable Example

The class location is also linked. Clicking on **114 Transportation Building** on the Timetable page, for example, would bring the user to a page with the floorplan of the first floor of the Transportation Building, with room 114 circled. (See Figure 14, which differs only in the room that is circled.) This gives a complete entry into the navigational system described above.

8.2 - Registration Materials Implementation

Programs Of Study

The *Programs of Study* unfortunately did not lend itself to automatic translation into hypertext. The document had a wide variety of notational conventions, as was perhaps to be expected from a document containing information gleaned from many sources. The document was also very long, and even the individual chapters were longer than is reasonable and customary for hypertext documents.

The document needed to be broken into many pieces, with appropriate cross-links to other sections. This required human intervention. In a hypertext document, it is meaningless to leave the phrase, "For more information, see the Office of Minority Affairs on page 59." I know of no computer program intelligent enough to translate "page 59" into the appropriate URL, especially when there can be two sections on page 59.

The manual intervention was not necessarily a large enough problem to make it a job not worth doing. By the time that this project was undertaken, campus awareness of the World-Wide Web was high enough that all of the colleges had information available on the web. I felt that it was quite possible that future versions of the *Programs of Study* would derive from colleges' online information instead of the other way around. I developed this version of the *Programs of Study* for the colleges to build upon; whether or not this will actually happen will not be known until long after I have graduated.

There was a great deal, fortunately, that could be automated. All courses that were in a regular format (i.e. the department abbreviations and three-digit call number) were automatically hyperlinked to the course description. In addition, room numbers coupled with building names were recognized automatically and linked to be navigational system described above.

Courses Catalog

To prove the concept, I typed in all of the course descriptions for the General Engineering Department, copying them from the university document called the *Courses Catalog.* The course descriptions looked like this:

```
221. Introduction to General Engineering Design. Fundamental
concepts in the analytical modelling, classical and computer-
based analysis and design of structural and machine
components and assemblies; external loads, internal forces
and displacements in statically determinate and indeterminate
configurations; kinematics of linkages, gears, and cams;
static forces in machines. Prerequisite: Theoretical and
Applied Mechanics 212 and 221, and Computer Science 101. 3
hours.
```

At first glance, this does not look like machine-readable text. However, further examination showed that there was actually great regularity in the entries. The first four characters are the three-digit course number and a period. From there to the next period is the course title. Until "`Prerequisite:`" is the course description, and from "`Prerequisite:`" to the next period is the prerequisite list. After that is the credit received for successful completion.

The prerequisite list contains names of departments that are fully spelled out, not abbreviated. This meant that I had to create a translation table to generate the course abbreviations (used in the URLs for the class schedule pages, as seen below). Otherwise, it was straightforward to generate the HTML that presented the course description in a fully hypertext fashion.

I then went to the Office of Facilities Planning and Management (OFPM) to attempt to get an electronic version of all of the engineering and math course descriptions. After some resistance on their part towards giving a document with some legal weight to a complete unknown, they relented and gave me a plain-text section of the *Courses Catalog* for engineering and math courses. I turned those all into hypertext documents.

This turned out to be just a prototype. At the time that I did this, I had no official standing, which made it difficult to get access to good information. Thus, at the request of higher authorities, I turned over my scripts to the Computing and Communications Services Office, to Mike Grady.

As the paper version of the *Courses Catalog* is only published on a two-year schedule, OFPM did not have any internal mechanism for incremental changes. Working with Mike, they instituted changes in the way they compiled the information so that the online version of the *Courses Catalog* could be updated more often than once every two years.

Mike also made some changes to the layout of the course description. He found that there were certain combinations of browsers and operating systems that would crash when confronted with very long documents. While the engineering classes could all be printed in one file, the Music Department had so many classes that they could not be listed on one page. An example of the course descriptions as Mike rearranged them is in Figure 28.

The Timetable information already existed online in a non-hypertext format through the `ph` facility. The `ph` database could be queried to give information about classes in a format similar to the following:

```
----------------------------------------
     name: music213 the history of music i
     text: fall92
         : prerequisite: music 110 or consent of instructor.
         : required of all music students.
         : 3 hours.
         : 05929 lect             1       m  w  f  2100  music bld
         : 05930 quiz       a     1        tu th   1180  music bld
         : 05931 quiz       b     9        tu th   1144  music bld
         : 05931 quiz       b     9        tu th   1144  music bld
         : 05932 quiz       c    10        tu th   1148  music bld
         : 05933 quiz       d     9          w  f  1184  music bld
         : 05934 quiz       e    11          w  f  1148  music bld
         : 05935 quiz       f     4        tu th   1161  music bld
   ----------------------------------------
```

I was able to write a script that parsed the record, saving important fields. The first word after "`name:`" is the department name and three-digit course number. The rest of the line is the course title. From "`prerequisite:`" to a period is a list of the prerequisites, with all words that are followed by three digits being the department, and the three-digit numbers being the associated course numbers. After the period of the prerequisites, a carriage return, and a colon is the credit given for the class. The rest of the lines pertain to the class times and locations, and are in a table based on column number.

Armed with this information, I could create am HTML document with the same information but formatted more nicely, and with a link to the course description.

Timetable

At the same time that I turned the *Courses Catalog* script over to Mike Grady, I gave him the script to parse the *Timetable*. As he was a full-time, permanent employee of the Computing and Communications Services Office (CCSO), he could access to the up-to-date database maintained by the University's administrative computing department. Mike had to rewrite the scripts to deal with the different input format, but access to the database allowed him to create a new hypertext version every morning. For the first time, students and faculty had access to current information instead of a snapshot of the information as it stood on the day the *Timetable* went to press. One area that had been particularly wanting was the instructor. Most teaching assignments were made after the *Timetable* went to press, so never made it into print. Now the teaching assignments are reflected almost immediately in the on-line version.

In addition, enough information is in the database that Mike was able to link the instructor to his or her ph entry. And, when I finished the maps and floorplans segment of the navigational system, I suggested to Mike how the *Timetable* could link to the navigational system. An example of the final version of the on-line *Timetable* is in Figure 29.

7.3 - Registration Materials Summary

Students now can examine all of the information they need to register from one source instead of three. The access is free to the student, while the paper versions of the *Courses Catalog* and *Programs of Study* must be purchased. The on-line information is more up-to-date than the paper documents. The on-line versions also benefit from being tightly coupled to information about the instructor and the navigational system.

9.0 - RESULTS AND FUTURE WORK

In the year and a half since this project began, the accesses to the University's top-level server have grown enormously. This chapter will describe measures of the success of this project and detail what further work is required.

9.1 - Results

When this project started, there were essentially no accesses to the top-level server, but as of November, 1995, there were routinely over 100,000 non-image accesses per week. While not all of these were to the navigation or course documents, a substantial percentage are. (See Appendix C for a historical data on accesses to the major subsystems.)

Similar growth in the campus as a whole has been clear. An effort was made by Graham Lawlor in the summer of 1995 to quantify the size of the campus-wide web. The web's success unfortunately meant that he was unsuccessful. His tools broke after indexing 40,000 documents from 276 different servers. This is a far cry from the hundreds documents and tens of servers in existence on the UIUC campus at the beginning of the project.

The University of Illinois Web has received recognition both on- and off-campus:

- The navigation system was featured on Yahoo's What's Cool page, resulting in a large amount of virtual visitors to the campus.

- The wheelchair access information, coupled with the maps, proved useful enough for the project to be recognized by Delta Sigma Omicron, the disabled student organization, with the Harold Scharper Service Award. Furthermore, DSO has committed to furthering its development.

- A paper on this project was accepted and presented at the 1994 Fall International World-Wide Web Conference. [1]

- A chapter was devoted to this project in a best-selling computer book [2].

The campus Web had a number of firsts:

- The campus Web had the second system to circle buildings on a map (after SUNY-Buffalo), the first to circle rooms on floorplans, the first to integrate floorplans and maps, the second to have a "virtual tour" (after Honolulu Community College), and the first to integrate maps/floorplans with a virtual tour. A number of people from off campus wrote to me asking for the scripts.

- The campus had the second cross-linked courses catalog and timetable (after St. Olaf's College), was the first to integrate courses catalog, timetable, and graduation requirements, and was the first to integrate the timetable with floorplans.

The UIUC Web resulted in high-level acceptance and involvement, as witnessed by:

- The navigational system has been deemed useful enough that the Office of Facilities Planning and Maintenance has committed to furthering its development.

- The first version of the hypertext courses catalog and timetable was so appealing that the Communication and Computing Services Office specifically requested responsibility for it.

- The Web version of the Timetable has been so successful and popular that the ph version is no longer maintained.
- A full-time position was created to continue managing the UIUC Web's framework. At the budget meeting of CCSO senior staff, the hiring of a good person for this job was accorded first priority.

There is also anecdotal evidence that people have incorporated the UIUC Web into their activities:

- Many home pages include links to the location of the office.

- General Engineering's Open House team used Transportation Building floorplans as an integral part of their information booth. There are also several places in the Digital Computer Laboratory where hardcopy of the floorplans is posted.

- I witnessed a student in a wheelchair carefully examining hardcopy of ion system, clearly trying to figure out how to get to class.

9.2 - Future Work

The work is by no means done. There are over 200 buildings on the UIUC campus, and only a fraction of the information associated with the buildings is on-line. The Delta Sigma Omicron has committed to furthering the wheelchair access information. The Office of Facilities Planning and Management has committed to making more floorplans and occupancy information available, and expressed interest in publishing information about building hours and classroom occupancy limits and equipment availability.

When file uploading capabilities are built into browsers, a mechanism for departments to be able to maintain their own information about room occupancy would be prudent. The Computing and Communications Services Office is currently investigating such a system.

The *Programs of Study* is updated every two years. Doing a wholesale translation from the paper version to a hypertext version can be hazardous to the health. It is hoped, both by the project author and by representatives of the Office of Publications, that at some point in the future, the paper version (if one continues to exist) will be derived from on-line information published by the appropriate functional units instead of vice-versa. It is doubtful that this will happen before the next version of the *Programs of Study* is published, but the opinion is that the version after that will originate on-line. The Office of Publications would be the natural organization to lead and coordinate such an effort.

The UIUC Web is clearly a compelling system, as seen by the number of accesses, the explicit recognition it has gained, the acceptance that it has received on-campus, and the anecdotal evidence. However, there is still a great deal of work to be done.

10.0 - SUMMARY AND CONCLUSIONS

This report has presented the background and work performed on a project that has developed important information resources for the University of Illinois' Web. Specifically, this report has reviewed the technical development of the Internet, including the various tools, protocols, and formats important to understanding the World-Wide Web. This report also has reviewed the development of campus Web resources prior to the start of this project.

The primary resources developed for this project are a set of indicies cataloging campus Web resources, documents used in course registration, and navigational aids. While development spanned several years, prototypes of the system components were in use by October of 1994. From this extended period of use, a number of conclusions can be drawn. On campus, this project has heightened awareness of and encouraced people to develop for the local Web. It has provided valuable resources to visitors, students, faculty, and staff. Off campus, it has provided publicity for the university, and allowed prospective students to better explore its offerings.

The World-Wide Web is a powerful channel for distributing rich hypermedia documents. However, even though some very significant development of Web tools was done at UIUC, there was very little material available on the UIUC Web before this project. Today, the UIUC Web is extremely rich and well-developed.

While certainly part of the explosion of materials on the UIUC Web is due to publicity about the World-Wide Web in traditional journalistic media, developing a good framework certainly aided its advance. Indices to campus resources provided users with the ability to find documents of interest, and served to encourage organizations to develop resources. Furthermore, skeletal home pages developed as part of this project have been picked up and enhanced by many organizations.

In addition to allowing others to better contribute to the Web, this project resulted in better distribution and presentation of important materials. No longer do students need to flip back and forth between three different, out-of-date documents to complete registration. The hyperlinking of the *Programs of Study*, *Courses Catalog*, and *Timetable* allows students to shift effortlessly between the documents. Students can now glide seamlessly all the way from graduation requirements to the location of a class. Furthermore, the electronic documents are updated more frequently than their paper counterparts, giving students much more accurate information.

This project also resulted in wide-spread availability of information that was not previously publically available. People can now routinely access interactive maps, floorplans, building occupancy, wheelchair access information, and even a virtual tour of the campus. This information is used on a day-to-day basis in place of written directions. This information is particularly useful to people with mobility restrictions, as shown by an award given by the disabled students' organization.

The project has also given the university a good deal of exposure. It has been the subject of a couple of publications and has been listed on Yahoo's *What's Cool* registry.

While the University still uses paper documents for many things, the trend towards electronic publication is already clear. This work helped to empower others to develop materials and developed useful resources. As such, it was a significant contributor to the UIUC Web.

APPENDIX A - who_is_in SOURCE CODE

This Appendix contains the source code to the perl script who_is_in, which is discussed in Section 7.2. No include files are needed. There are no restrictions on use, modification, or redistribution of this code. An electronic version of this code is available at http://www.uiuc.edu/navigation/tools/who_is_in.

```
#!/usr/local/bin/perl

# This is a script that tells who is in a given room, based on a data
# file named <bldg>.who.data with the following format:
# <dept>          <room>          <URL>
# e.g. the file tb.who.data has the following line:
# ge              105       http://uxh.cso.uiuc.edu/~gedept/ge/directory/faculty/Goldberg.html
# (The department name is to update the information properly, Talbot, for example, has
# people from TAM and Aero both in it, and we want to be able to update the
# Aero people separately.)

# This script parses the query string and data file to figure out where the
# person's URL really is, then does a relocate to pass that info.


################################################################################
#
# Set up knownledge about the world outside this script
################################################################################
#
# (Note that pond has links set up so that vixen's paths will work)

# debug code -- Mike Grady 4/2/95
#print "Content-type: text/html\n\n";
#print "<h1>Testing</h1>\n";
#print "Just testing and logging<p>\n";
#open (LOG, ">/tmp/whereis");
```

```
#select (LOG);

$DATADIR = "/var/info/www/docs/navigation/data";

$BLDGURL = "http://www.uiuc.edu/navigation/buildings";
$COEURL = "http://www.cen.uiuc.edu/COE-Info/coe.top.html";
$UIUCURL = "http://www.uiuc.edu/";
$WEBMASTER = "<i><a
href=http://www.webfoot.com/ducky.home.html>ducky@uiuc.edu</a></i>\n";
$DOCROOT = "/var/info/www/docs";

$NAVROOT = "$DOCROOT/navigation";

$TMPDIR = "$NAVROOT/tmp";
push (@INC, "$NAVROOT/tools");
# Load in the long/short name translation table
require "initbldg.pl";

##############################################################################
#
# Parse the query string
##############################################################################
#
$query = $ENV{'QUERY_STRING'};

if ($query =~ /room=(\d*\w?)/) {
        $room = $1;
        }

if ($query =~ /bldg=([\w\+]*)/) {
        $bldg = $1;
        }
else {
        print "Content-type: text/html\n\n";
        print "<h1>Error</h1>\n";
        print "Sorry, this script needs a building to work!<p>\n";
        print "Usage:who_is_in?room=<i>room</i>&bldg=<i>bldg</i>.<p>";
        exit;
        }
```

```
$bldg =~ tr/+/ /;

################################################################################
#
# The options on the pulldown menu in locator.html
# have to be spelled out fully, sigh, before they get passed over to this script.
# (This is supposed to change in ?HTML+.)  This checks to see
# if it is a short name ($bldg) or a long name ($longbldg), or neither

# commented out following -- Mike Grady 4/2/95
#print "bldg is $bldg, longbldg is $longbldg<br>\n";
if (!$longbldg{$bldg}) {
        foreach $key (keys %longbldg) {
                if ($longbldg{$key} eq $bldg) {
                        $bldg = $key;
                        break;
                        }
                }
        if (!$bldg) {
                print "Content-type: text/html\n\n";
                print "<title>Error</title>";
                print "<h1>Error</h1>";
                print "<hr>Sorry, that building was not recognized.  Try again.\n";
                }
        }


# $room =~ tr/A-Z/a-z/;
# $bldg =~ tr/A-Z/a-z/;

################################################################################
#
# Open the data file
################################################################################
#
if (!open(DATAFILE, "$DATADIR/$bldg.who_is_in.data") ) {
        print "Content-type: text/html\n\n";
        print "<h1>Sorry</h1>\n";
        print "Occupancy information is not available for $longbldg{$bldg} yet.<br>\n";
        exit;
        }

# If $room is specified, do the redirection
if ($room) {
        &findRoom();
        }
# If $room is not specified, create an ordered, ASCII
# list of the building's contents
else {
        &makeBldgDir();
        }

####################################################################################
#
# Look through the data file to find who is where
####################################################################################
#
sub findRoom {
        # Parse the data file to get the room information
        while (<DATAFILE>) {
                #     department room    URL
                if (/^([\w\-]*)\s+(\w*)\s+([\w*:\/\.~\'\-\#\+\?\%]*)\s+(.*)/) {
                        # print "URL is $3, room is $2, dept is $1\n";
                        if ($room eq $2) {
                                push(@munged, $3 . "@" . $4);
                                break;
                                }
                        }
                }

        # If there is only one person in the room, redirect us to their URL
        if ($#munged == 0) {

                $mungee = pop(@munged);
                ($url, $person) = split(/@/, $mungee,2);
                if ($url) {
                        print "Location: $url\n\n";
                        }
        }

    # If there is more than one person in the room, make a list
    # of their URLs.
    elsif ($#munged > 0) {
        print "Content-type: text/html\n\n";
        print "<h1>Room $room</h1>\n";
        print "There is more than one entry for Room $room.  Your options are:\n";
        print "<ul>\n";
        while ($#munged >= 0) {
            $mungee = pop(@munged);
            ($url, $person) = split(/@/, $mungee,2);
            print "<li><a href=$url>$person</a><br>\n";
            }
        print "</ul>\n";

        }
    else {
        print "Content-type: text/html\n\n";
        print "<h1>Deserted</h1>\n";
        print "We don't know who is in room $room, sorry.\n";
        }

    }

##############################################################################
#
# makeBldgDir makes an ordered list of a building's contents
##############################################################################
#
sub makeBldgDir {
    while (<DATAFILE>) {
        #   dept      room                    URL              name
        if (/^([\w\-]*)\s*(\w*)\s*([\w*:\/\.~\'\-\#\+\?\%]*)\s*(.*)\s*([\w\-\.\,]*)/) {
            # Note that the room has to be included, since
            # there are a number of rooms marked "Classroom" and
            # "Bathroom"
            $roomURL{$4."@@@".$2} = $2."@@@".$3;
            }
```

```
}


# Print out header
print "Content-type: text/html\n\n";
# GOTTA figure out how to include this stuff!
print "<title>Directory for $longbldg{$bldg}</title>\n";
print "<h1>Directory for $longbldg{$bldg}</h1>\n";
print "This list is in a relatively stupid alphabetical ordering.\n";
print "If you can't find the person, lab, or office you were looking\n";
print "for, try using your browser's search function.";
print "<hr>\n";
print "<pre>\n";
print "Room	Occupant\n\n";

foreach $key sort(keys %roomURL) {
	($name, $room) = split('@@@', $key);
	($room, $url) = split('@@@', $roomURL{$key});

	print "<a href=http://www.uiuc.edu/cgi-bin/where_is?bldg=$bldg&room=$room>$room</a>   <a href=$url>$name</a><br>\n";
	}


print "</pre><p>\n<hr>\n";
print "Go to <a href=$BLDGURL/$bldg.top.html>$longbldg{$bldg}</a><br>\n";
print "Go to <a href=$COEURL>College of Engineering Home Page</a><br>";
print "Go to <a href=$UIUCURL>University of Illinois Home Page</a><br>\n";
print "\n<hr>\n$WEBMASTER";
}
```

APPENDIX B - where_is SOURCE CODE

This Appendix contains the source code to the `perl` script `where_is`, which is discussed in Section 7.2. No include files are needed, but a circling routine is required. (The circling routine was done by Carlos Pero; I do not have the right to redistribute it.) There are no restrictions on use, modification, or redistribution of this code. An electronic version of this code is available at `http://www.uiuc.edu/navigation/tools/where_is`.

```
#!/usr/local/bin/perl
#
# Script to take in building abbreviation and room number, parse, locate
# map file, and read in coordinates.

# Where_is.pl  by Carlos A. Pero
# 10/14 hacked on by Ducky Sherwood
# 10/26 hacked on some more so that if a building hasn't been
# floorplanned, but does exist on the campus map, it will
# return the building.
# 10/30 Lowercase the building and room now.
# 10/30 fixed a bug - DUH, Duck!  where I was -split-ing based
# on the room number, duh, what happens if there is a coordinate
# with the same room number, duh?
# 11/5 fixed two bugs: eliminated case sensitivity and also
# fixed it so that the separator between the two points
# in the ismap file does not HAVE to be a tab.
# 12/27 Subroutinified it in preparation for librarification
# 6/13/95 Changed the contact information at the bottom of the page.
# 11/15/95 Changed the wording slightly, alphabetized the locator
#           bar better - ducky

# -----------------------------------------------------------
# Required files:
```

```
#
# where_is.gd   (gd binary)
# ----------------------------------------------------------

# Set variables

# pond.cso directories
# Ooops, won't run on pond 'cuz no where_is.gd,
# should fix that someday...
$BINDIR = "/usr/local/etc/httpd/cgi-bin";
$DOCROOT = "/www";

# vixen.cso directories
$BINDIR = "/var/info/www/httpd/cgi-bin";
$DOCROOT = "/var/info/www/docs";

$NAVROOT = "$DOCROOT/navigation";
$TMPDIR = "$NAVROOT/tmp";
$SHORTTMP = "/navigation/tmp";
$CAMPUSMAP = "north_campus";
$MAPDIR= "$NAVROOT/maps";
$GIFDIR = "$NAVROOT/floorplans";
$HTMLDIR = "$NAVROOT/floorplans";
$ISMAPDIR = "$NAVROOT/ismaps";
$DATADIR = "$NAVROOT/data";
$PID = $$;   # I don't think I did, but somebody had getppid here...  -kds
$WEBMASTER = "ducky@uiuc.edu";
$MAILPROG = "/usr/lib/sendmail";
push (@INC, "$NAVROOT/tools");



# Print out a content-type for HTTP/1.0 compatibility
print "Content-type: text/html\n\n";

# Load in the long/short name translation table
require "initbldg.pl";

&cleanOldGIFs;
```

```
&readArgs;
&parseOptions;
&long2short;

# Load in the tree of what map a building is on
# This needs to happen late so that the where_is of parent works...
require "mapparent.pl";

&whichGIF;
&readIsmap;
&calcCoords;

if ($room) {
        &showRoom;
        }
elsif ($shelf) {
        &showShelf;
        }
else {
        &showBldg;
        }
&writeFooter;

#####################################################################################
#
# Need to clean up old tmp files; can't remove them near the
# end of the script since Mosaic reads the GIF *after* the
# end of the script.
#####################################################################################
#
sub cleanOldGIFs {
        if (! opendir(TMP, $TMPDIR) ) {
                print "<title>Error</title>\n";
                print "<h1>Error</h1>\n";
                print "Could not read directory $TMPDIR.\n";
                exit;
                }
        @oldfiles =  readdir(TMP);
        foreach $oldgif (@oldfiles) {
                # This *should* be easier - see about putting everything on one line...
```

```
                    # If there are any .gifs lying around that are more than a
                    # day old, zap them.
                    if (-f "$TMPDIR/$oldgif") {
                              if ((-M "$TMPDIR/$oldgif") > .002)  {
                                        unlink("$TMPDIR/$oldgif");
                                        }
                              }
          }
          }


######################################################################################
#
# Get the input, either as GET or POST
######################################################################################
#
sub readArgs {
          if ($ENV{'REQUEST_METHOD'} eq "POST") {
                    read(STDIN, $buffer, $ENV{'CONTENT_LENGTH'});
                    }
          else {
                    $buffer = $ENV{'QUERY_STRING'};
                    }

          }

######################################################################################
#
# Parse out the room and building
######################################################################################
#
sub parseOptions {
          @pairs = split(/&/, $buffer);

          foreach $pair (@pairs)
          {
                    ($name, $value) = split(/=/, $pair);

                    # Un-Webify plus signs and %-encoding
```

```
        $value =~ tr/+/ /;
        $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg;

        # Strip out any bad, bad characters
        $value =~ s/[!\|;\/]//g;
        $value =~ s/\s*$//;

        # Uncomment for debugging purposes
        # print "$name is $value<P>";

        if ($name eq "bldg") {
                $bldg = $value;
                }
        elsif ($name eq "room") {
                $room = $value;
                }
        elsif ($name eq "library") {
                $library = $value;
                }
        elsif ($name eq "shelf") {
                $shelf = $value;
                }
        elsif ($name eq "street") {
                $street = $value;
                }
        elsif ($name eq "callno") {
                $callno = $value;
                }
}


if ($callno) {
        &callno2shelf;
        }

####
# Check that a building was specified
if (!$bldg) {
        # This will have to change for the main lib, which has
        # multiple libraries in one building
```

```
                if ($library) {
                        $bldg = $library;
                        }
                else {
                        print "<h1>Error</h1>";
                        print "<hr>Sorry, no building was specified.  Try again.\n";
                        exit;
                        }
                }


        }


##########################################################################
#
# The options on the pulldown menu on the north campus page
# have to be spelled out fully, sigh, before they get passed over to this script.
# (This is supposed to change in ?HTML+.)  This checks to see
# if it is a short name ($bldg) or a long name ($longbldg), or neither
##########################################################################
#
sub long2short {
        if (!$longbldg{$bldg}) {
                foreach $key (keys %longbldg) {
                        if ($longbldg{$key} eq $bldg) {
                                $bldg = $key;
                                break;
                                }
                        }
                if (!$bldg) {
                        print "Content-type: text/html\n\n";
                        print "<title>Error</title>";
                        print "<h1>Error</h1>";
                        print "<hr>Sorry, that building was not recognized.  Try again.\n";
                        }
                }
        }
```

```
#######################################################################################
#
# Find line with building name and room number in the map file
# Construct filename template (i.e.  tb.1f )
# If there is a room, then this is a floorplan, otherwise it
# is a building on the campus map
#######################################################################################
#
sub whichGIF {
        if ($room) {

                # We need to deal with basement floors, since they
                # are not usually listed as "014".



                # DCL has basement rooms called things like "L440"
                if (substr($room, 0, 1) =~ /L/i) {
                        $FLOOR = "b";
                        }
                else {

                        ($digitroom = $room) =~ s/[a-z]//ig;                    # turn e.g. 308A into 308
                        if ($digitroom < 100) {
                                $FLOOR = "b";
                                }
                        else {
                                $FLOOR = substr($room, 0, 1);
                                }
                        }

                $FILENAME = "$bldg.$FLOOR"."f";
                }

        elsif ($shelf) {
                # We need to deal with basement floors, since they
                # are not usually listed as "014".
                ($digitroom = $shelf) =~ s/[a-z]//ig;     # turn e.g. 308A into 308
                $digitroom =~ s/#[0-9]+//;                # split off the shelf number
                if ($digitroom < 100) {
```

```
            $FLOOR = "b";
            }
        else {
            $FLOOR = substr($shelf, 0, 1);
            }

        $FILENAME = "lib_$bldg.$FLOOR"."f";
        }
    else {
        $FILENAME = "$parent{$bldg}";
        }




    # Figure out if this building has been mapped yet.  If it hasn't, then
    # use do a building search instead of a room search.
    if (! -f "$ISMAPDIR/$FILENAME.ismap") {
    $room = "";
        if ($parent{$bldg}) {
            $FILENAME = "$parent{$bldg}";
            }
        else {
            print "<title>Error</title>\n";

            if ($ENV{'HTTP_USER_AGENT'} eq "NCSA Mosaic(tm) Version 2.0.0a9 for
Windows") {
                print "<h1>WinMosaic Bug!</h1>\n";
                print "You are using Mosaic for Windows 2.0.0a9, which has a bug that
screws some links (especially from the Timetable!)\n";
                print "It passes the building number properly, but does not pass the room
number properly; sorry.  :-(</b><p>\n";
                print "You need to <a
href=http://www.ncsa.uiuc.edu/SDG/Software/WinMosaic/HomePage.html>";
                print "go get the latest release</a> of Mosaic for Windows. (2.0.0b9
should work)\n";
                }
            elsif ($ENV{'HTTP_USER_AGENT'} eq "IBM WebExplorer/1.0") {

                print "<h1>IBM WebExplorer Bug!</h1>\n";
                print "You are using IBM WebExplorer, which has a bug that screws up
```

```
some links!\n";
                print "It only passes the first word of a building's name; sorry.  :-
(</b><p>\n";

                print "You can work around it by using the abbreviated name listed
below.<p>\n";
                }
            else {
                print "<h1>Ooops!</h1>\n";

                print "Ooops!  The webmaster has neglected to enter what map
<b>$bldg</b> is on.  \n";
                print "<p>Either that or you've made an error in your request,\n";
                print "or you've hit the <b>icky Windows Mosaic 2.0.0a9
bug!</b><p>\n";
                print "Note: NetManage Chameleon Mosaic+ 4.03 might also have a
bug in it.\n";

                }

            print "Mail has been automatically sent to $WEBMASTER about the
error.<p>\n";
            print "If you got here via the locator, things are supposed to work.\n";
            print "If you came here via a link, the link might be bad.  The proper syntax is
something like<p>\n";
            print "<ul><tt>http://www.uiuc.edu/cgi-
bin/where_is?bldg=nel&amp;room=101</tt></ul><p>\n";
            print "Legal buildings are as follows:\n<pre>\n";
            foreach $key sort(keys(%longbldg)) {
                print   "       $key                        $longbldg{$key}\n";
                }
            print "</pre>\n";



            if (!open (MAIL, "|$MAILPROG $WEBMASTER")) {
                print "Furthermore, mail is somehow broken.  Could you \n";
                print "please send email to <a
href=mailto:$WEBMASTER>$WEBMASTER</a> saying that \n";
```

```
                                print "$bldg has no map parent?  Thanks a bunch.\n";
                                }
                        else
                                {
                                print MAIL "Subject: $bldg missing mapparent\n";
                                print MAIL "Building $bldg (longbldg = $longbldg{$bldg}) has no
mapparent!\n";
                                print MAIL "\n\nQUERY STRING IS\n\n\t$buffer\n\n";
                                print MAIL "\nHTTP_REFERER is $ENV{'HTTP_REFERER'}, AGENT
is $ENV{'HTTP_USER_AGENT'}.\n\n";
                                print MAIL "REMOTE HOST is $ENV{'REMOTE_HOST'},
SERVER_NAME is $ENV{'SERVER_NAME'}\n\n\n";
                                print MAIL "Add $bldg to $NAVROOT/tools/mapparent.pl.\n";
                                close (MAIL);
                                }
                        exit;
                        }
                }
        }


################################################################################
#
# Figure out what the coordinates are for the room
################################################################################
#
sub readIsmap {
        # Read in the .ismap file
        if (!open (MAPFILE, "$ISMAPDIR/$FILENAME.ismap")) {
                print "<title>Error</title>\n";
                print "<h1>Error</h1>\n";
                print "<hr>\nFile $ISMAPDIR/$FILENAME.ismap could not be read.\n";
                print "bldg = $bldg, parent = $parent{$bldg}, longbldg is $longbldg{$bldg}\n";
                print "Please tell the webmaster, <i>$WEBMASTER</i>.\n";
                }
        while (<MAPFILE>) {
                chop;
                # print $foo;
                $foo = $_;
                # $foo = s/\s*/ /g;         # not sure if this is needed; just being safe
                ($type, $line) = split(/rect\s/, $foo);
                # print $line;
                push (@lines, $line);
                }
        close (MAPFILE);

        if ($room) {
                @match = grep(/bldg=$bldg/i && /room=$room\b/i, @lines);
                }
        elsif ($shelf) {
                @match = grep(/library=$library/i && /shelf=$shelf\b/i, @lines);
                }
        elsif (grep(/maps\/$bldg.html/i, @lines)) {
                @match= grep(/maps\/$bldg.html/i, @lines);
                }
        else {
                @match= grep(/buildings\/$bldg.top.html/i, @lines);
                }



        # Parse out coordinates
        if ($room) {
                ($foo, $coords) = split(/=$room\b/i, $match[0]);
                }
        elsif ($shelf) {
                ($foo, $coords) = split(/=$shelf\b/i, $match[0]);
                }
        else {
                ($foo, $coords) = split(/$bldg.[\.top]{0,4}html/i, $match[0]);
                }
        $coords =~ s/^\s*//;
        ($coord1, $coord2) = split(/\s/, $coords);

        # print "Match[0] is $match[0]\n";
        # print "coords $coords split to $coord1, $coord2\n";
        }

###############################################################################
```

```
# Calculate the coordinates for the gd binary
##############################################################################
sub calcCoords {
    # Do calculations for gd binary
    ($x1, $y1) = split(/,/, $coord1);
    ($x2, $y2) = split(/,/, $coord2);


    $cx = int(($x1+$x2)/2);
    $cy = int(($y1+$y2)/2);
    $dx = $x2 - $x1;
    $dy = $y2 - $y1;
    if ($dy > $dx) {
        $wh = 1.2* int($dy);
        }
    else {
        $wh = 1.2* int($dx);
        }

    }


##############################################################################
#
sub showBldg {
    # print a title and initial heading
    # print "exec: $BINDIR/where_is.gd $MAPDIR/$parent{$bldg} $TMPDIR/$PID $cx $cy
$wh\n";
    `$BINDIR/where_is.gd $MAPDIR/$FILENAME $TMPDIR/$PID $cx $cy $wh`;
    print "<Head><Title>Location of $longbldg{$bldg}</Title></Head>\n";
    print "<Body><H1>Location of $longbldg{$bldg}</H1>\n";
    print "Click on any building/area that is in blue to get further information\n";
    print "about that building.<br>\n";
    print "You can use the menu bar below to find a building.<p>\n";
    if ($FILENAME =~ /all_campus/) {
        print "If you want to find a building that is inside one of the blue rectangles, (not on the
menu bar)\n";
        print "use the <a href=/navigation/locator.html>Campus Locator</a>.<p>\n";
        }
    &writeLocatorMenu;
    print "<P>\n";
    print "<a href=/cgi-bin/imagemap/$FILENAME>\n";
    print "<IMG SRC=\"$SHORTTMP/$PID.gif\" ismap></a><p>\n";

    }

##########################################################################
#
sub showRoom {
    # print "exec:\n\t$BINDIR/where_is.gd $GIFDIR/$FILENAME $TMPDIR/$PID $cx $cy
$wh";
    `$BINDIR/where_is.gd $GIFDIR/$FILENAME $TMPDIR/$PID $cx $cy $wh`;

    # Print a title and initial heading
    print "<Head><Title>Location of Room $room in $longbldg{$bldg}</Title></Head>\n";
    print "<Body><H1>Location of Room $room in $longbldg{$bldg}</H1>\n";
    print "<P>\n";
    print "<a href=/cgi-bin/imagemap/$bldg"."_".$FLOOR."f>\n";
    print "<IMG SRC=\"$SHORTTMP/$PID.gif\" ismap></a><p>\n";

    print "<a href=/cgi-bin/where_is?bldg=$bldg>Where am I on campus?</a><br>\n";
    print "<p>Go to <a href=/navigation/buildings/$bldg.top.html>$longbldg{$bldg} lobby</a>
(main building information page)<p>\n";
    }

##########################################################################
#
sub showShelf {
    # @@@ should really fix up so that lib_grainger.2f.gif doesn't have to be linked to
grainger.2f.gif...
    # print "exec:\n\t$BINDIR/where_is.gd $GIFDIR/$FILENAME $TMPDIR/$PID $cx $cy
$wh";
    `$BINDIR/where_is.gd $GIFDIR/$FILENAME $TMPDIR/$PID $cx $cy $wh`;

    # Print a title and initial heading
    print "<Head><Title>Location of Shelf $shelf in $longbldg{$bldg}</Title></Head>\n";
    print "<Body><H1>Location of Shelf $shelf in $longbldg{$bldg}</H1>\n";
    print "<P>\n";
```

```
    print "<a href=/cgi-bin/imagemap/$bldg"."_".$FLOOR."f>\n";
    print "<IMG SRC=\"$SHORTTMP/$PID.gif\" ismap></a><p>\n";

    print "<a href=/cgi-bin/where_is?bldg=$bldg>Where am I on campus?</a><br>\n";
    print "<p>Go to <a href=/navigation/buildings/$bldg.top.html>$longbldg{$bldg}</a> main
information page<p>\n";
    }
##############################################################################
#
# Write out the footer
##############################################################################
#
sub writeFooter {

    # Figure out which map this belongs to
    if (!($room || $shelf)) {
        $bldg = $parent{$bldg};
        }

    while ($parent{$bldg}) {
        print $anchor{$parent{$bldg}};
        $bldg = $parent{$bldg};
        }

    print "<hr>\n";

    print "This page was made possible in part by a grant from <a
href=http://www.uiuc.edu/ccso/>CCSO</a>.\n";

    print "<H5>On-the-fly graphics by Carlos A. Pero (c-pero@uiuc.edu)\n";
    print "using the <a href=http://siva.cshl.org/gd/gd.html>gd 1.0</a> library</h5>\n";
    print "<P>";
    print "Surrounding scripts by\n";
    print "<a
href=http://www.webfoot.com/ducky.home.html?Maps><i>ducky@webfoot.com</i></a>,\n";
    print "maintained by\n";
    print "<a
href=mailto:webmaster@www.uiuc.edu><i>webmaster@www.uiuc.edu</i></a>.\n";
    }
```

```
################################################################################
#
# Write out the locator menu
################################################################################
#
sub writeLocatorMenu {
        print "<form method=\"GET\" action=\"http://www.uiuc.edu/cgi-bin/where_is\">\n";
        print "<select name=\"bldg\">\n";
        foreach $key sort(keys(%parent)) {
                if ($parent{$key} eq $FILENAME) {
                        # print "<option>$longbldg{$key}\n";
                        push(@foundbldg, $longbldg{$key});
                        }
                }
        foreach $fndbldg sort(@foundbldg) {
                print "<option>$fndbldg\n";
                }
        print "</select>\n";
        print "<input type=\"submit\" value=\"Where is it? \"><p>\n";
        }

################################################################################
#
# This routine takes a call number (in $callno), reads a data file, and figures
# out which shelf it is on.
################################################################################
#
sub callno2shelf {
        $LIBRARY_DATA = "$DATADIR/$library.callnos";
        # At some point, there needs to be some logic to figure out which file to
        # open (based on which library I'm looking at).
        if (!open (LIBDATAFILE, "$LIBRARY_DATA") ) {
                        print "<title>Error</title>\n";
                        print "<h1>Error</h1>\n";
                        print "Can't read $LIBRARY_DATA<br>\n";
                        exit;
                        }

        # The format of the data file is pretty important.  It goes
        # Room_number#shelf_number  starting_call_num-ending_call_num     starting_call_num-
ending_call_num
        # (There are two sets of start and end because there are two "faces" to each shelf.)
        # Note that the fields MUST be delimited by tabs.  This is because there are
        # sometimes spaces in the call numbers, e.g. "622.05MINI, VOL. 31"
        while (<LIBDATAFILE>) {
                if (/^([0-9]*\-[0-9]+)\t+([0-9][0-9][0-9][,\.\w]*)-[0-9[0-9][0-9][ ,\.\w]*\t+[0-9][0-9][0-
9][\.\w]*-([0-9[0-9][0-9][\.\w]*)/) {
                        $shelf = $1;
                        $callstart = $2;
                        $callend = $3;
                        # Note that this depends upon @ not showing up in any call numbers.
                        $shelf{"$callstart@$callend"} = $shelf;
                        }
                }

        # The next line is just in case the above list is not sorted by call number
        @shelfarray = sort ( keys ( %shelf));
        # Pop one out.  (Note that because pop takes off the end,
        # we get the largest number first and the smallest number
        # last.
        ($callstart, $callend) = split('@', pop(@shelfarray));

        # I wanted to do a simple binary search, but there are real
        # troubles if a callnumber that falls in the cracks between
        # the shelves (e.g. 551.49N42G) is entered.  So do it
        # the stupid way, and just wait until the
        while (($callno lt $callstart) && ($#shelfarray >= 0)) {
                ($callstart, $callend) = split('@', pop(@shelfarray));
                }

        # There can be cases where a call number is requested
        # that is between the shelves as listed.  (This can happen
        # easily if a book from another library is requested.)
        # In this case, a meltdown is probably not in order - just
        # a simple warning message.
        if ($callno gt $callstart) {
                if ($callno gt $callend) {
                        print "<h2>Warning</h2>\n";
```

```
                    print "That call number is between shelves!\n";
                    print "You might be looking in the wrong library,\n";
                    print "the call number might be wrong,\n";
                    print "or the shelf list might be out of date.\n";
                    print "Here is a shelf that is close.\n";
                    }
          }

$shelf = $shelf{"$callstart@$callend"};
}
```

APPENDIX C - ACCESS DATA

Here are the number of accesses to the registration and navigation subsystems.

Time Period	Navigation System	Courses Catalog and Timetable	Programs of Study
Aug 1994		1,036	
Sep 1994		7,420	
Oct 1994	8,529	11,858	
Nov 1994	22,118	17,028	
Dec 1994	17,948	9,540	
Jan 1995	35,358	25,462	
Feb 1995	33,310	19,495	
Mar 1995	37,737	128,616	
Apr 1995	N/A	N/A	
May 1995	26,623	40,005	
Jun 1995	N/A	N/A	
Jul 1995	N/A	N/A	
Aug 1995	14,837	59,897	4,640
Sep 1995	18,994	42,885	6,642
Oct 1995	22,585	201,085	7,527

Table 1 - Access Counts

Data is unfortunately not available for all periods. Note that these counts do not reflect images; access counts would be substantially higher if images were included.

LIST OF REFERENCES

[1] Sherwood, K. D. [1995] “A Campus-Wide Information System”,*HTML and CGI Unleashed*, John December, editor,Sams.Net Publishing, Indianapolis.

[2] Sherwood, K. D. [1994] “The University of Illinois College of Engineering Web”, *Proceedings of the Second International WWW Conference*.