Designing, Implementing, and Maintaining Digital Library Services and Collections with MyLibrary

Eric Lease Morgan, et al.

Designing, Implementing, and Maintaining Digital Library Services and Collections with MyLibrary

by Eric Lease Morgan, et al.

This manual is a part of system of files called MyLibrary, a database-driven website application intended to facilitate digital library collections and services.

This manual is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This manual is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this manual if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

For more information about this MyLibrary and this manual, see: http://dewey.library.nd.edu/mylibrary/ [http://dewey.library.nd.edu/mylibrary/] .

Table of Contents

List of Tables

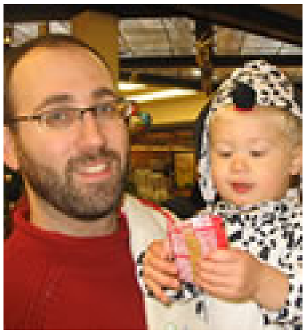
Part I. Contributors

Table of Contents

Chapter 1. Contributors to MyLibrary Manual

Table 1.1. Contributors to MyLibrary Manual

Name	Photo	Bio	Articles
Eric Morgan University of Notre Dame		Bio	- Foreword - Information Architecture - Implementation and Maintenance - MyLibrary Tutorial
Alison Aldrich Wright State University Libraries		Bio	- MyLibrary and Campus Portal: a Case Study
Vishwam Annam Wright State University Libraries		Bio	- What are Relational Databases and Why Should I Care? - MyLibrary and Campus Portal: a Case Study
Martin Courtois Information Technology Assistance Center Kansas State University		Bio	- Low-cost Recording of Usability Tests
Nora Dimmock University of Rochester			- Paper Prototyping

Name	Photo	Bio	Articles
Robert Fox University of Notre Dame		Bio	- Perl API
Terry Huttenlock Buswell Library Wheaton College		Bio	- Usability Testing: a Key to User-centered Designs
Megan Johnson Appalachian State University		Bio	- Focus Group Interviews
Leslie Johnston University of Virginia Library		Bio	- The Importance of Content Standards in Digital Libraries - Usability Case Study: University of Virginia
Peter Karman		Bio	- What are Indexers and Why Should I Care?

Name	Photo	Bio	Articles
Harold Kirkwood Management and Economics Library Purdue University		Bio	• Usability Case Study: Purdue University Libraries
Tom Lehman University of Notre Dame		Bio	• Facets and Terms • Surveys
Shelley McKibbon Dalhousie University Libraries			• Card Sorting
Keith Morgan North Carolina State University Libraries		Bio	• Pioneering Portals: A History Of MyLibrary@NCState
Terry Nikkel Dalhousie University Libraries			• Card Sorting
Tod Olson Integrated Library Systems The University of Chicago		Bio	• What is XML and Why Should I Care?

Name	Photo	Bio	Articles
Brenda Reeb University of Rochester		Bio	• Communicating Usability Results
Michael Yunkin University of Nevada, Las Vegas Libraries		Bio	• Attracting Users

Bios

Alison Aldrich

Alison Aldrich is a health sciences reference librarian at Wright State University. She also serves on the Libraries' web team. Alison has a Master of Science in Information degree from the University of Michigan and is pursuing a degree in public health at Wright State.

Vishwam Annam

Vishwam Annam is working as a web developer at Wright State University Libraries. He is involved in programming, researching and teaching in academic libraries for more than six years. He has received Master of Science degree in Computer Science from Western KY University.

Vishwam is interested in investigating the way library users interact with information technology, both as they affect technological development and as they are affected by technology. As a teacher of archival technologies and information literacy, he is interested in understanding how users make sense of the information they find. To that end he is also interested in Human Computer Interaction from a design standpoint and hopes that his work will be of use to software engineers, Librarians, and Web designers.

Marty Courtois

Marty Courtois has worked as a reference and instruction librarian at California State University - Long Beach and University of Illinois at Chicago, and served as head of reference at Illinois Institute of Technology. At Michigan State University, he worked as a science librarian and coordinator of database services. He served as biological sciences librarian at University of Tennessee. At George Washington University, Marty worked as the engineering/computer science librarian and as web development librarian. At GW, and in his current role as computer information specialist at Kansas State University, Marty has been involved in designing and conducting web usability tests.

Robert Fox

Robert Fox is a Sr. Programmer / Analyst for the University Libraries of Notre Dame and works for Eric Morgan in the Digital Access and Information Architecture Department. Rob has been working in library systems administration and programming positions since 1998. In his current position, he has worked on a range of digital library projects which include campus portal syndication of library content, a virtual new acquisitions service, page turner software, and the Perl modules for MyLibrary 3.0, and is a columnist for the journal 'OCLC Systems and Services' published by Emerald. Rob is also an Oracle OCP and one of his favorite topics is database programming and design.

Terry Huttenlock

Terry Huttenlock is an Assistant Professor of Library Science and the Head of Systems and Technological Services for Buswell Library at Wheaton College, Wheaton Illinois. She worked for fifteen years as a computer systems analyst in corporate before deciding to change careers and return to school to obtain her MLS. During her studies she became involved in user needs research which sparked her current interest in usability. Currently she is a doctoral candidate in instructional technology at Northern Illinois University working on her thesis that deals with a design element that could be effective in our online learning environments.

Megan Johnson

Megan Johnson is the Web Services Librarian at Belk Library and Information Commons, Appalachian State Library in Boone, NC. She got her Master of Library and Information Studies University of North Carolina at Greensboro, 1993. Currently she is working on a second masters degree in Educational Media.

Leslie Johnston

Leslie Johnston is the Head of Digital Access Services at the University of Virginia Library, where she manages digital library program components supporting the collection, management, and delivery of digital content. Previously, she served as the Head of Instructional Technology and Library Information Systems at the Harvard Design School, as the Academic Technology Specialist for Art for the Stanford University Libraries, and as Database Specialist for the Getty Research Institute. Ms. Johnston has also been active in the museum automation community, working for various museums, teaching courses in museum studies, editing the journal Spectra, and serving on the board of the Museum Computer Network.

Peter Karman

In addition to his work designing search systems and web sites, Peter Karman is a father, husband, and musician. He lives in St Paul, Minnesota.

Hal P. Kirkwood Jr.

Professor Kirkwood is the coordinator of the Instruction Section for the Management and Economics Library. He is responsible for the development of the Management and Economics Library Web site as well as collection development in strategic management, operations management, marketing, quantitative methods, and MIS. His research interests are in teaching business information research, usability testing, information architecture, and business database comparison. Professor Kirkwood has been awarded the Moriarty Award for Excellence in Librarianship by the Purdue University Libraries and the Class of 1922 Helping Students Learn Award by Purdue University.

Tom Lehman

Tom Lehman is a member of the Digital Access and Information Architecture Dept. in the University of Notre Dame Libraries, where he works on the Libraries' intranet and extranet and helps maintain the University-wide search engine.

Eric Lease Morgan

Eric Lease Morgan is the Head of the Digital Access and Information Architecture Department at the University Libraries of Notre Dame. He considers himself to be a librarian first and a computer user second. His professional goal is to discover new ways to use computers to provide better library service. Some of his more well-known investigations and implementations include MyLibrary, the Alex Catalogue of Electronic Texts, and the Mr. Serials Process. An advocate for open source software and open access publishing, Eric has been freely distributing his software and publications for years before the terms "open source" and "open access" were coined. In his copious spare time, Eric can be seen folding defective floppy disks into intricate origami flora and fauna. Eric also hosts his own Internet domain, infomotions.com.

Keith Morgan

Keith Morgan is Client Services Librarian in the Digital Libraries Initiative Department at North Carolina State. He was part of the original design and development team for MyLibrary@NCState and although not related to Eric Morgan, has always been pleased that Eric credited him with coming up with the term "MyLibrary."

Tod Olson

Tod Olson is a Programmer/Analyst at the University of Chicago Library, and has been working in library computing for over a decade. He is best known for his work on the Chopin Early Editions collection of piano scores and speaking about Greenstone Digital Library Software. Tod holds an MSLIS from the University of Illinois at Urbana-Champaign.

Brenda Reeb

Brenda Reeb launched the usability program at the University of Rochester River Campus Libraries in 2001. The usability team conducts usability tests on several digital library projects using a variety of testing methods. She is the listowner of USABILITY4LIB, a forum for usability work in a library environment. Brenda offers regional and national usability workshops, and is often an invited speaker on library technologies and user interface design.

In addition to her user interface work, Brenda serves as the director of the Management Library for the Simon School of Business at the University of Rochester, and she supervises the government information and microtext units. A native rural Midwesterner, Brenda earned her BS from Loyola University of Chicago and her MLS from Simmons College.

Michael Yunkin

Michael Yunkin is the Web Content/Metadata Manager at UNLV Libraries in fabulous Las Vegas. He is also the creator and administrator of Digiwik: The Digitization Wiki.

Part II. Foreword

Table of Contents

Chapter 2. Foreword to Designing, Implementing, and Maintaining Digital Library Services and Collections with MyLibrary

Eric Lease Morgan, University of Notre Dame

Who should read this manual and why

The purpose of this manual is to outline the principles and processes necessary to implement digital library collections and services. It uses MyLibrary as an example.

The manual is intended to be read by library administrators who need to know what and how many resources to allocate to a digital library. It is intended to be read by librarians who are responsible for collecting and organizing content as well as ensuring the system's usability. The manual is intended to be read by systems administrators who are in charge of providing the technical infrastructure for the system. Last but not least, it is intended for programers who will use the underlying Perl API to provide services against the collection.

In a library setting, especially an academic library setting, these four groups of people -- library administrators, librarians, systems administrators, and programmers -- have more things in common than differences. They all have ultimate goals surrounding the collection, organization, preservation, and dissemination of data and information for the purposes of facilitating learning, teaching, and research. This set of common goals binds the group and provides a sense of direction. The group's complementary sets of skills, knowledge, and expertise ensure the system's functionality and usability. Everybody has something to offer, and everybody's talents are required to make the implementation of a digital library a success.

Implementing digital libraries

The implementation of a digital library, including a MyLibrary implementation, can be distilled into a number of tasks:

1. Create a group of people who have the necessary skills, outlined above.
2. Create an information architecture for the system - This "blueprint" will answer difficult questions regarding the system's users, content, and context. By the end of this process you will have articulated the expressed needs of the primary audience, a collection development policy, and a mission statement for the system. This process is described in more detail in the Chapter 3
3. Decide on an ontology for logically organizing your content - MyLibrary employs a facet/term approach to classification. Facets are broad headings and terms are specific. Example facet/term combinations might include Subjects/Astronomy, Formats/Books, or Audiences/Freshman. Any number of combinations can be created in MyLibrary, and any number of them can be associated with any particular information resource. This is described in more detail in Chapter 4
4. Collect and organize your content - MyLibrary uses Dublin Core elements combined with the facets and terms for this purpose. You might acquire your content programatically by importing

MARC records or harvesting content from OAI-PMH repositories. You will probably want to supplement the collection/organization process with manual data entry techniques. In either case, MyLibrary comes with example scripts illustrating how to accomplish these goals.

5. Create services against the collection - At the very least you will want to provide browsable and/or searchable interfaces to your collection. Your browsable interface will almost undoubtedly mirror your facet/term classification system. Your searchable interface may query the underlying MyLibrary database directly or access indexes created against the system. If your content is dynamic, then you might want to provide a What's New service. You may want to provide a user-driven, customizable interface. The services you implement will be guided by the blueprints you outlined in your information architecture. Programs/scripts demonstrating these sorts of services are described in the MyLibrary Tutorial.

6. Do usability testing - While your implementation may very well be functional, it might not be usable. List six or seven tasks your digital library is intended to allow users to accomplish. Randomly ask users to do the tasks and watch what happens. If the users accomplish the tasks successfully then you are a winner, if not, then you must redesign and re-implement. Various techniques for doing usability testing are described in Chapter 6.

7. Communicate - Throughout this entire process you need share your progress with everybody around you. Consider creating a Digital Library Advisory Group whose membership consists of a minority of librarians. These people can serve as an ongoing focus group, and they continually answer the question, "Is the system headed in the right direction?" Inside the library ask questions such as, "Does this system solve more problems than it creates?" Marketing and promotion are essential before, during, and after the system's implementation. Creating your digital library should not be a well-kept secret.

8. Repeat - Libraries are not about books, and digital libraries are not about websites. Instead, libraries to one degree or another are about the never-ending processes of collecting, organizing, preserving, and disseminating data and information. Audiences, content, and contexts continually evolve, necessitating constant re-examination and revision. Go to Step #1.

MyLibrary is open source software

Open source software is about community. It is about creating computing solutions to problems, sharing those solutions, improving upon them, and giving these solutions back to the community. Open source software is about pooling resources, building relationships, and improving the way things are done. These processes are very similar to the processes of librarianship. Consequently open source software and librarianship have direct relationships to each other, and MyLibrary is a particular example of that relationship. Because of this relationship the implementation of digital libraries using open source software, specifically with MyLibrary, allows librarians to have direct control over their hardware and software as opposed to the other way around. No vendors. No contracts. No annual fees. No scapegoats.

This does not mean open source software is without expenses. The largest expense will be time. Time spent answering the questions of information architecture, communicating the ongoing development of the system, and the maintenance of the content. The second largest expense will be the development of services against the collection -- the writing of computer programs against the system by computer programmers. This "expense" is really an investment, an investment in the personnel necessary to implement and maintain digital library collections and services. Comparatively, the final expense is almost trivial, the expense of purchasing a computer to host the system.

MyLibrary is written in and distributed as open source software. This means the software is free -- free as a "free" kitten. Not only are there no monetary costs to acquire the software, but it also means you are free to modify the software in any way you desire. You are just not allowed to redistribute the software or anything based on it under a license that is more limiting than the original GNU Public License.

For more information about MyLibrary visit the website at http://dewey.library.nd.edu/mylibrary/, and consider participating in the MyLibrary community by subscribing to the mailing list.

See you on the 'Net.

Part III. Information architecture

Table of Contents

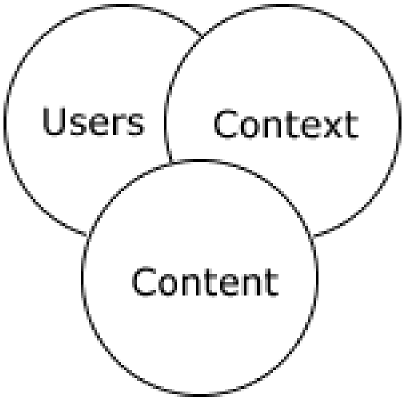
Chapter 3. First Principles of Information Architecture: "On your Mark. Get set. Go!" not "Fire, and then Aim."

Eric Lease Morgan , University of Notre Dame

Introduction

At its core, information architecture is about users, context, and content. By answering questions regarding these issues your MyLibrary implementation will not only be functional. It will be understandable to your intended audience, serve a meaningful purpose, and contain relevant content. Information architecture is the result of a planning process. It is about "On your mark. Get set. Go!" not "Fire, and then aim." This essay elaborates on these ideas and outlines some of things you need to think about as you begin to implement any information system, not just MyLibrary.

A definition



Information architecture is often illustrated using a Venn diagram depicting three interlocking circles representing users, context, and content. *Users* are the intended audience of an information system, *context* is the reason the systems exists, and *content* is the data/information the system has to communicate. For good information architecture to take place, a concrete understanding of an information system's audience, purpose, and data/information is necessary. This is like the architecture of buildings, where an understanding of who is going to live there, what the building is for, and what it will contain must be outlined before construction can begin.

At the risk of pushing the metaphor too far, the result of information architecture is a "blueprint illustrating the framework" which you will fill with content, organize with controlled vocabularies, hang site-wide navigation, and make browsable as well as searchable. If you do this with an eye to satisfying the expressed needs and desires of your users as well as your hosting institution, you will end up building something usable (not just functional), and they will come.

Users

The first step in designing your information architecture is answering questions regarding users. You need to define the primary audience of your information system, build relationships with them, and learn what they need and desire.

Defining your information system's primary audience is easier than you may think. In a private university like Notre Dame, the primary audience includes the University's students, faculty, and staff. The needs of these people take precedence over the needs of the general public, alumni, or scholars from other institutions. There are limited resources (time and money) allotted to the implementation of your information system, and it is not possible to be all things to all people. Consequently, you need to prioritize and decide to whom, primarily, you are going to cater your service. At a public university, the audience may be broader, including the general public, especially the public of the immediate area or region. In a public library, the primary audience may be area residents. In special and school libraries, the answers to these questions will seem almost obvious.

After defining who your audience is, you need to establish inter-personal relationships with them. No, you don't have to become their best buddy, but you do need to build rapport to learn their expressed needs and desires. You need to learn and, more importantly, understand the challenges and difficulties they are having when it comes to doing their work. I'm sure you can create a long list of their challenges and difficulties, but since you are not them you can not prioritize which of the challenges and difficulties are the ones they need addressed. By building relationships with your primary audience you will learn these priorities and be able to focus your resources on making them easier to accomplish.

There are many ways to build relationships and learn of your audience's priorities. Surveys are the first thing that come to mind. They are relatively inexpensive. They can touch large numbers of people, and they are good for answering "what" types of questions. "What is your age?" "What do you like and what do you dislike about our present information system?" "If you could change one thing, what would it be?" The answers to survey questions often need to be short and succinct; few people are going to give you a lot of detail while answering survey questions. The results of surveys usually manifest themselves numerically and then get converted into graphs. Along the lines of surveys are log file analysis. By looking at the statistics captured by your staff as well as your present information systems, you will get an idea of what your audience uses. People will often say one thing and act differently. Log files help put this behavior into perspective.

The other side of surveys are focus group interviews, structured communication sessions used to learn about your audience's feelings. When compared to surveys, focus group interviews require a greater degree of interpersonal skills on the part of a facilitator. They touch fewer people than surveys and therefore are often times seen as more expensive. On the other hand, focus group interviews answer questions surveys don't answer, specifically "why" questions. "Why do you like this service as opposed to another?" "Why do you think it is important to for us to implement such and such feature?" "Why do you spend your time working in this particular manner?" Just like surveys, the focus group interview process ranges from the simplistic to the complex. It can be as simple as a one-on-one chat over coffee, or it can be as elaborate as a meeting of six to twelve homogeneous people who answer questions in a moderated setting by a professional facilitator.

In conclusion, in order to learn about your audience's needs and desires, consider issuing one or more surveys first and following up with sets of focus group interviews second. This process will enable you to validate the survey's conclusions and learn why people answered the survey the way they did.

Context

The next step in articulating an information architecture is to answer questions regarding context. What is the purpose of your information system, how does it fit within the totality of your institution's products and services, and what sorts of resources (time and money) are allotted to the system's development and maintenance?

Your information system will not exist in a vacuum. It will be a reflection of its hosting institution, and in order for the information system to reflect well you will need to know the goals and priorities of your institution. For example, you need to know the purpose of the hosting institution. What problems is it trying to solve? How can your information system be expected to contribute to the solutions? Look to your institution's mission statement for answers. Here at Notre Dame the library's role is to help the students, faculty and staff of the University community do their learning, teaching, and scholarship. The role of our website (and our MyLibrary implementation) mirrors the purpose of the University Libraries: to help facilitate learning and teaching, to assist in scholarship, to supplement access to collections and service, and to facilitate communication.

The context of your information system will also be tempered by the amount of resources allotted to its development and maintenance. These resources take the form of time, money, hardware, software, people, and expertise. The implementation and ongoing maintenance of your information system will require a diverse set of skills. None of which are necessarily more important than the other. The people with the necessary skills include subject experts, leaders of people, graphic designers, people who can mark up texts, knowledge workers who can organize content, usability experts, marketers, programmers, and systems administrators. The amount of time and energy these sorts of people can bring to the implementation of your information system is directly proportional to what your information system will enable people to do and do well. When the Web began a little more than fifteen years ago people's expectations were low, but with the growing size and diversity of the Web, people's expectations have matured, and consequently so does the need to allocate more resources to your implementation.

Defining the purpose of your information system and articulating what resources will be spent on its development is the second step in the creation of your information architecture. Do not set your goals too high lest you set yourself up for failure. Determine the importance of your information system compared to the other products and services you offer, and allocate your resources accordingly.

Content

The third step in the creation of your information architecture is defining what content it will contain. This is akin to articulating a collection development policy.

Not even Google provides access to the totality of the world's content, and there is no reason to expect you to fill this role. Instead, focus on the answers regarding users and content to define the scope of your content. Ask yourself, "What are the strengths of my institution?" "To what degree does my collection need to be comprehensive, authoritative, up-to-date, written in a particular language, presented in an aesthetically pleasing manner, etc.?" In other words, create a list of guidelines that your information resources need to embody in order to be a part of your collection. Just because a particular information resource is about a particular subject does not necessarily mean it is a good candidate.

When the University Libraries of Notre Dame re-created its website using MyLibrary, we decided the content would not be very much different from the content of traditional, physical libraries. It contains tools to access bibliographic information, access to digital library services and collections, instructions for pedagogy, and last but not least, access to people who can help with all these processes -- librarians. The website is not designed to be a comprehensive list of resources. Instead, it is designed to highlight the most significant resources and provide starting points for learning and research. The content of the website is very much like the content of traditional library pathfinders.

Summary

Information architecture is about answering difficult questions regarding users, context, and content. It is not possible to be everything to everybody, therefore you need to define who your primary audience is. *Users*. Your information system is a part of a larger institution, therefore it behooves you to make sure the system fits into the institution's goals and objectives. *Context*. The world of information is too large for any system to embody, and therefore you need to limit the scope of your collection. *Content*.

Once you answer the questions regarding users, context, and content, write down the answers. Use them as guidelines for a specific period of time (at least one year), and then regularly revisit the guidelines.

On your mark. Get set. Go. Not, fire and then aim.

--

Eric Lease Morgan

October 12, 2005

Chapter 4. Facets and Terms in MyLibrary

Introduction

The purpose of this essay is to describe how facets and terms work in MyLibrary. After reading this, you should have an idea of how to proceed in classifying your content as you implement MyLibrary at your institution.

Background

In its previous incarnation, v.2, MyLibrary had three groupings or categories into which information resources could be placed [1]:

- Ready reference shelf
- Databases
- Etexts and ejournals

This was based on the way resources tended to be grouped or categorized in the electronic resources gateways of the time.

When Notre Dame librarians looked at MyLibrary as a possible vehicle for implementing the redesigned library website, a common reaction was that the "three piles" model was too restrictive. "Databases" ended up being a miscellaneous category for resources that didn't fit easily into either of the other categories. The categories were hard coded into the software and could not be easily changed.

When the MyLibrary software was rewritten as part the Notre Dame Libraries' website redesign, in place of the three fixed categories, a completely generalized system of classification was implemented: the Facet and Term system.

Facets and Terms in MyLibrary

In MyLibrary v.3, Facets are seen as categories and Terms as instances of the category. For example, if "Subjects" were established as a MyLibrary facet, possible terms for that facet might be: "Biology," "Chemistry," or "Chemical and Biomolecular Engineering." If "Formats" were set up as a MyLibrary facet, possible terms might be: "Sound recordings," "Maps," "Microforms."

This Facet/Term approach is powerful and flexible. Using facets and terms, you can classify almost anything you would want to include in MyLibrary, for example Librarians or Services.

At Notre Dame we've found ways to use facets and terms that aren't immediately apparent. We've created a facet called "Access" that describes whether a resource is licensed, free, or installed on a particular workstation. A facet called "Flags" (the word "Flags" is used in the "on/off" computer sense) lets us create terms that can be used to identify all the resources with a particular condition for some special treatment in MyLibrary. We've used this facet to identify cross reference records in MyLibrary that provide access to variant titles, and also to identify records for customized MetaLib QuickSearches appearing on various subject pages, to allow specialized placement of the QuickSearches on those pages.

[1] MyLibrary v.2 had additional categories for other types of resources; e.g. "Librarians," "Quicksearches," "Library links."

Unlike some other classification systems, which have multiple layers of hierarchy; e.g. Plants # Trees # Oaks # White oak, the Facet/Term system only has those two levels. However with the use of inverted phrases one can achieve a sort of three-level hierarchy at the cost of using some slightly stilted language. For example:

Facet	**Languages**
Term	German
Term	Greek, Ancient
Term	Greek, Koine
Term	Greek, Modern
Term	Hebrew

Creating Facets and Terms for MyLibrary

The power and flexibility of the Facet/Term system in MyLibrary comes at a cost. Rather than assigning terms from a system-provided thesaurus, you will need to create your own facets and terms, or adopt existing an existing vocabulary.

To create Facets and Terms for your MyLibrary implementation, you will need to know who your users are and how they look for and use information. Are they chiefly undergraduates who will need guidance as to what resources they should be using first? Some way to identify and clearly designate resources as "Core" or "Best place to start your research" may be helpful to that group. Are they graduate students and faculty who are familiar with the literature in their area and mainly need easy, direct access to it? Listing resources by topic, format, etc. as well as alphabetically may be important for them. Most likely you will need to provide for a variety of users.

As you set up your facet and term structure, terminology will be critical. Will the term "Index" or "Database" mean anything to your students? Card sorting, described elsewhere in this manual, can be a useful tool for finding out what terms make sense to your users.

At Notre Dame, for the most part, we based the terms in the "Subjects" facet on departmental names, as we believed our users are more likely to be familiar with those terms.

In addition to knowing your users, you will need to know your content--the resources you will be making available through MyLibrary. While users will typically begin their search looking for information using a topical approach, topic is not the only useful approach to resources. Other types of information users might look for might include:

- type of reference tool (e.g. dictionary)
- newspapers published in a particular country (format or genre, place)
- language of the material
- genre: drama or poem

Some facets and terms will be suggested by the nature of the resources you will be adding to MyLibrary. For example:

a. What the resource is about; e.g. topic or subject. For example:

Facet:	Subjects

Term:	Biology
Term:	Chemistry

b. Physical characteristics or format of the resource. For example:

Facet:	Formats
Term:	Audio recordings
Term:	Photographs and images
Term:	Video recordings

c. Type of reference tool. For example:

Facet:	Reference tools
Term:	Bibliographies
Term:	Dictionaries
Term:	Encyclopedias

d. Type of content. For example:

Facet:	Genre
Term:	Drama
Term:	Poetry
Term:	Tests

e. Language of the resource. For example:

Facet:	Languages
Term:	English
Term:	French
Term:	German

f. Audience for the resource. For example:

Facet:	Audience
Term:	Children
Term:	Faculty
Term:	Undergraduate students

Several things to consider as you create your vocabulary:

- Facets should be distinct and unambiguous. Writing definitions for each facet will help. Plan on testing your vocabulary. It should be obvious which terms go with which facet. It is probably clear that the term "Biology" belongs under the facet "Subject," but it may not be as clear whether the term "Conference proceedings" belongs in the facet "Format" or "Genre" or something else entirely. Time spent up front clarifying these issues will save time and prevent confusion later.

- Principle of warrant. Unless your MyLibrary implementation is intended to be comprehensive there is no need for your vocabulary to cover all possible cases. If your literature collection is primarily in European languages, there is no need for the term "Quechua" in your "Languages" facet. It is easy to add terms later as needed.

- Think about how you plan to apply the vocabulary terms to resources. A number of electronic resources combine elements of very different print resources. Will you want to assign terms to a resource for each aspect of a resource (includes a dictionary, directory, biographies, index, etc.) or find a term that applies to the resource as a whole? Experience has led us from the former practice to the latter.

- Document your decisions
- Set up a process to consider and implement changes to the vocabulary.

Conclusion

As previously mentioned, the MyLibrary Facet/Term approach is powerful and flexible. Using it makes it possible to organize resources and use them in a variety of ways. For example, the vocabulary (and software) can be used to create web pages, printed guides, or channels to a campus portal application.

As another example, if you assigned the facet/term combinations:

- Language::French
- Resource type::Dictionary
- Subject::Medieval studies

to resources in your MyLibrary implementation, you could use MyLibrary to make these pages:

- all Dictionaries, subdivided by language
- all French dictionaries
- all French language resources
- a Medieval Studies page with a section for dictionaries

Using Facets and Terms effectively requires careful analysis, knowledge of your users and your content, persistence and, perhaps most importantly, plenty of communication and cooperation with other library staff.

If these requirements are met, the results should be well worth the effort.

Part IV. Content standards

Table of Contents

Chapter 5. The Importance of Content Standards in Digital Libraries

Leslie Johnston, University of Virginia Library

The use of content standards and the documentation of standards used is a vital strategy for both access to digital collections and their preservation. The use of metadata standards ensures the discoverability and use of digital collections across multiple contexts, and consistent adherence to format standards extends the life cycle of digital files as technologies evolve. The use of format and metadata standards are equally important aspects of content best practice in building digital collections.

Metadata for Digital Objects

Many use the term "metadata" to refer solely to digital objects, making it seem a different undertaking from "cataloging." The creation of metadata is not a conceptually new activity; the cataloging of physical holdings is the creation of metadata. There are additional needs in terms of metadata that document the technology associated with the objects that are used to manage large, disparate, and possibly distributed collections of digital files, but much of what is needed is much the same as for physical objects - authoritative description of the title, creator(s), subject(s), and geographical coverage, and documentation of holdings, location, and status.

Metadata may be either embedded in the digital object, or exist externally to the digital object. Metadata is generally separated into four broad categories:

- *Descriptive metadata*: Information that documents the intellectual content and context.
- *Administrative metadata*: Information regarding creation date of the digital resource, copyright, use rights, etc.
- *Technical metadata*: Information that documents the attributes of an object, such as the digital capture process, media format, file size, and pixel dimensions. In some schemes, technical metadata is part of the administrative metadata.
- *Structural metadata*: Information that describes the relationships between files that might make up an object, or the relationship between objects that make up a larger conceptual whole.

As an example, for a digital image, one might have the following metadata recorded:

Descriptive:

- **title:** Bear Dance, Preparing for a Bear Hunt
- **artist:** Catlin, George
- **creation date:** 1835-1837
- **medium:** oil
- **physical description:** 19 5/8 x 27 1/2 in.
- **type:** artwork ; painting
- **subject (AAT):** landscapes (representations)
- **subject (AAT):** Native Americans
- **subject (LCSH):** Indians of North America
- **tribe:** Western Sioux ; Lakota
- **identifier (SAAM):** nnnn.nnn.nnn
- **credit:** Smithsonian American Art Museum, Gift of Mrs. Joseph Harrison, Jr.

Administrative:

- **persistent identifier:** uva-lib:nnnnn
- **creator:** University of Virginia Library
- **publisher:** University of Virginia Library
- **access:** Publicly accessible
- **copyright:** copyright 2004, by the Rector and Visitors of the University of Virginia
- **creation date:** 20040527

Technical:

- **file size:** 4.1 Mb
- **mime type:** image/tiff
- **compression:** none
- **color space:** RGB color
- **image width:** 1475
- **image length:** 961
- **source X:** 72
- **source Y:** 72
- **bits per sample:** 8
- **samples per pixel:** 3

Structural: (documents the location to access the various files that make up the object)

- **thumbnail:** http://server/directory/object_thumbnail
- **screen size:** http://server/directory/object_screen
- **max size:** http://server/directory/object_max
- **descriptive metadata:** http://server/directory/object_descmeta
- **administrative metadata:** http://server/directory/object_adminmeta

Metadata Best Practices: Encoding Rules and Controlled Vocabularies

Encoding standards are codifications of the practice of organizing data. This ranges from data dictionaries that describe local fields or elements and the standards for their use, to international standards for the creation of shareable metadata. Best practices for encoding range from a consistent record structure, to the consistent use of the data structure in a record, to the consistent use of controlled vocabularies in those records, and down to the consistent encoding of the characters in those records.

Examples of encoding standards:

- Structure of records:

  - Categories for the Description of Works of Art (CDWA): http://www.getty.edu/research/conducting_research/standards/cdwa/

  - Data Documentation initiative: http://www.icpsr.umich.edu/DDI/

  - Dublin Core: http://dublincore.org/

  - Encoded Archival Description (EAD): http://www.loc.gov/ead/

  - Federal Geographic Data Committee (FGDC): http://www.fgdc.gov/standards/standards.html

  - MARC: http://www.loc.gov/marc/
  - Metadata Encoding and Transmission Standard (METS): http://www.loc.gov/standards/mets/
  - Metadata Object Description Schema (MODS): http://www.loc.gov/standards/mods/
  - Text Encoding Initiative (TEI): http://www.tei-c.org/
  - PREMIS Preservation metadata: http://www.loc.gov/standards/premis/
  - VRA Core: http://www.vraweb.org/vracore3.htm
- Cataloging standards and use of data fields:
  - Anglo-American Cataloging Rules (AACR2): http://www.aacr2.org/
  - Cataloging Cultural Objects (CCO): http://www.vraweb.org/ccoweb/
- Character encoding:
  - Unicode: http://www.unicode.org/

Content data for some elements, such as the subject element, may be selected from a "controlled vocabulary," a limited set of consistently used and carefully defined terms. Using terminology from a controlled vocabulary ensures consistency and can improve the quality of search results, and may also reduce the likelihood of spelling errors when recording metadata. The description of each element indicates whether content should be selected from a controlled vocabulary, if possible.

Examples of controlled vocabularies:

- Web Thesaurus Compendium: http://www.ipsi.fraunhofer.de/~lutes/thesoecd.html (a compendium of controlled vocabularies)
- Index to Organism Names: http://www.biosis.org.uk/ion/search.htm
- Getty Research Institute Data Standards and Guidelines: http://www.getty.edu/research/conducting_research/standards/
- Getty Art and Architecture Thesaurus: http://shiva.pub.getty.edu/aat_browser/
- Getty Thesaurus of Graphic Names: http://shiva.pub.getty.edu/tgn_browser/
- Getty Union List of Artists Names: http://shiva.pub.getty.edu/ulan_browser/
- ICONCLASS: http://www.iconclass.nl/
- Library of Congress Authorities: http://authorities.loc.gov/
- Medical Subject Headings (MeSH): http://www.nlm.nih.gov/mesh/meshhome.html
- NASA Thesaurus : http://www.sti.nasa.gov/thesfrm1.htm
- Nomenclatural Glossary for Zoology: http://scientific.thomson.com/support/products/zr/zoological-glossary/
- *The Revised Nomenclature for Museum Cataloging, A Revised and Expanded Version of Robert G. Chenhall's System for Classifying Man-Made Objects* by James R. Blackaby, Patricia Greeno, and

the Nomenclature Committee. Published by American Association for State and Local History, 1988.

- SPECTRUM Terminology: http://www.mda.org.uk/spectrum-terminology/
- Thesaurus of Graphic Materials I: http://lcweb.loc.gov/rr/print/tgm1/toc.html
- Thesaurus of Graphic Materials II: http://lcweb.loc.gov/rr/print/tgm2/

Digital Media File Formats

The long-term utility of digital media files remains a major unknown. As data files are created and collected, their creators and stewards must take pains to ensure that the content will be accessible for as long as possible. Using standard file formats and documenting their use is the best guarantee of that longevity. Research the de facto standards in use in the community, decide what formats are most appropriate for your collections, use the formats consistently, document your content creation practices, and include information about the digitization process in the technical metadata that accompanies your digital objects. Consistent and well-documented practices will mean that future migrations to new formats (and preservation migration of formats is a certainty) are more likely to be successful.

There is much discussion in the community of the importance of using non-proprietary file formats. It is crucial to the long-term survival of digital content that it be created using file formats that can be migrated into new formats when necessary. Proprietary formats can often be migrated, but files in proprietary formats may be inaccessible once the software that created them has disappeared from the market. Migrations from non-proprietary or open standard formats can more likely be carried out without the cooperation of a software vendor, since the formats are publicly defined. This is not to say that you must avoid all content in proprietary formats or never use in it your operations - formats like Word or PDF (until fully replaced by PDF/A) are often unavoidable when accepting deposited content or necessary to ensure broad usability of files. When proprietary formats are unavoidable, documenting their use and avoiding proprietary features that complicate data migration are the best practices to follow. This article will not endorse or recommend any particular formats.

The following sites offer discussions and recommendations of technical formats for different types of digital content:

- Collaborative Digitization Project Digital Audio Best Practices: http://www.cdpheritage.org/digital/audio/documents/CDPDABP_1-2.pdf
- Digital Information Preservation: http://www.rlg.org/ArchTF/tfadi.index.htm
- DLF Benchmark for Faithful Digital Reproductions of Monographs and Serials: http://www.diglib.org/standards/bmarkfin.htm
- Guidelines for Computer File Types, Interchange Formats and Information Standards: http://www.collectionscanada.ca/06/0612/061204_e.html
- JPEG 2000 in Archives and Libraries: http://j2karclib.info/
- Library of Congress, Digital Formats for Content Reproductions: http://lcweb2.loc.gov/ammem/formats.html
- Motion Picture Experts Group (MPEG): http://www.chiariglione.org/mpeg/
- NINCH Guide to Good Practice in the Digital Representation and Management of Cultural Heritage Materials: http://www.nyu.edu/its/humanities/ninchguide/

- NISO Framework of Guidance for Building Good Digital Collections: http://www.niso.org/framework/Framework2.html

- Digital Media File Types: Survey of Common Formats (NIST): http://www.itl.nist.gov/div895/isis/filetypes.html

- PADI Format Standards: http://www.nla.gov.au/padi/topics/452.html

- PADI Formats and media: http://www.nla.gov.au/padi/topics/44.html

- PDF/A: http://www.aiim.org/standards.asp?ID=25013

- Society of Motion Picture and television Engineers (SMPTE) Standards: http://www.smpte.org/smpte_store/standards/

- Sustainability of Digital Formats: Planning for Library of Congress Collections: http://www.digitalpreservation.gov/formats/index.shtml

- VRA Guides to Quality in Visual Resource Imaging: http://www.rlg.org/legacy/visguides/

Interoperability and Aggregation

Interoperability is the ability of two or more information systems to exchange and use information. Metadata interoperability is highly dependent upon the ability to map identical or similar elements of data structures. The simplest strategy for interoperability is for each system to employ similar data structures and similar or identical encoding semantics and controlled vocabularies, as with MARC records, AACR2, and controlled vocabularies in the creation of library records. In the reality of the digital library world, metadata is created by a myriad of repositories using a myriad of data structures, semantics, and vocabularies and taxonomies (community-based or local), all valid for their context and environment.

The most difficult task in metadata interoperability is semantic and taxonomic compatibility. Full metadata interoperability across diverse systems with diverse content is impossible. Achieving even limited interoperability between systems requires coordination, most often achieved through a common mapping between the fields or elements that each system uses to organize metadata. A "map" documents the correspondences between similar data fields or elements in different systems or standards. When exchanging information, each system can recognize the mapping between fields or elements and translate between them.

Best practices demand consistent utilization of encoding rules and practices (e.g. AACR2 or CCO [Cataloging Cultural Objects]), and standardized vocabularies (e.g., the Getty Art & Architecture Thesaurus, Library of Congress Subject Headings, or those developed by specialized communities), as well as documentation of the vocabularies in use for each field or element. Digital library programs should develop and maintain "data dictionaries" that document all fields or elements in all local systems and the semantics, taxonomies, and vocabularies used in all instances. Also vital is documentation between local practice and other standards and systems called a "crosswalk," documenting the mappings between local practice and MARC, Dublin Core, VRA Core (Visual Resources Association), Text Encoding Initiative (TEI), and Encoded Archival Description (EAD).

There are interoperability concerns not just for metadata, but for the files that make up the content of the digital objects. Some institutions use TIFF, some use JPEG 2000, and some use Mr. Sid. TEI has passed through a number of stages of development, and some institutions use TEIxLite with others use TEI P4. Formats change over time. TIFF has had many versions. There is a new TEI profile under development. JPEG 2000 is a relatively new standard. PDF/A is a developing open version of PDF. There is no right or wrong; each institution picks what is appropriate for its needs at the time of collection building.

There are many best practices to keep in mind for format interoperability:

- Review the standards used in the community. Broad acceptance of a format generally translates to broad support for a format across many systems.

- Use format standards consistently. Do not use TIFF for one project and JPEG 2000 for another project unless there is some functionality required by a project that translates to the need for a different format.

- Document your use. Record which formats you use for all types of collections, as well as the details of your use. Do not just record that you use TIFF - record that you capture 32-bit color masters at 600 dpi, saving the files in the TIFF format. As another example, do not just record that you use TEI - record any local standards for the use of the TEI elements or local extensions that you may use, which will effect the ability of other systems to parse your TEI files.

- Include technical metadata in the metadata for your objects. This improves the likelihood that other systems will be able to recognize and use your files. At a minimum, document the MIME type in a standardized way.

- Validate, refresh and migrate. Periodically review files on live disk and offline media to confirm that the files are uncorrupted and usable. Refresh offline storage media regularly. If you are switching to a new format - such as from Mr. Sid to JPEG 2000 - migrate your existing collections in addition to changing your new production. This will simplify your delivery systems and simplify any future format migrations.

While these practices are essential in any local digital collection building initiative, the need is magnified in the sharing or aggregation of metadata and collections. Many institutions make the metadata representing their digital collections available for harvesting and aggregation through the Open Archives Initiative Protocol for Metadata Harvesting (http://www.openarchives.org/). When records from disparate institutions are aggregated in services such as OAIster (http://oaister.umdl.umich.edu/o/oaister/) or the Collaborative Digitization Project (http://www.cdpheritage.org/), it is a boon to the user to have so many resources available through a single search interface. It is also a challenge to the user because some institutions use MARC and some use ContentDM and some use EAD and some use TEI, and so on. How have those institutions recorded their metadata, and how have they mapped it? Will a search for "ocean" find all the applicable content? While every institution should use the standards that are most appropriate for their collections, all institutions should pay attention to community practices in the use of record formats, encoding practices, controlled vocabularies, and mapping and crosswalking between standards to better ensure that their collections are shareable. One truth has become self-evident in this area: the richer the metadata - descriptive, administrative, technical and structural - and the more standard the formats used, the better the chances of interoperability.

Professional Organizations and Authoritative Sites on Digital Collection Best Practices

The following sites link to professional organizations, projects, and reports that, in addition to those in the section above, provide a wealth of experience and information for any institution planning projects to create, manage, and deliver collections of digital content.

- Association of Moving Image Archivists: http://www.amianet.org/publication/publication.html

- Association for Recorded Sound Collections (ARSC): http://www.arsc-audio.org/

- Collaborative Digitization Project Digital Toolbox: http://www.cdpheritage.org/digital/index.cfm

- Digital Library Federation: http://www.diglib.org/

- International Research on Permanent Authentic Records in Electronic Systems (InterPARES): http://www.interpares.org/
- Library of Congress Digital Collections and Programs: http://www.loc.gov/library/libarch-digital.html
- Museum Computer Network, Standards: http://www.mcn.edu/resources/sigstandards/
- National Information Standards Organization: http:www.niso.org/
- Northeast Document Conservation Center Handbook for Digital Projects: http://www.nedcc.org/digital/dighome.htm
- OCLC Research: http://www.oclc.org/research/default.htm
- RLG Guides and Tools: http://www.rlg.org/en/page.php?Page_ID=555
- Society of American Archivists: http://www.archivists.org/
- Strategies for Building Digital Collections: http://www.clir.org/pubs/reports/pub101/contents.html
- Technical Advisory Service for Images: http://www.tasi.ac.uk/advice/advice.html
- Visual Resources Association, Resources: http://www.vraweb.org/resources.html

Part V. User-centered design

Table of Contents

Chapter 6. Usability Testing: a Key to User-centered Designs

Terry Huttenlock, Buswell Library, Wheaton College

We continually hear the words usability and user-centered being used, but do we really understand that these terms mean? In Donald Norman's popular book, *The Design of Everyday Things*, usability is based on how people interact with objects. His illustrations stress the need to make products and systems usable and understandable. The key to design is to make sure the user can figure out what to do and also understand what is going on. This includes appropriate clues so the user can make the appropriate actions -- the appropriate actions should be intuitive. In the design of web environments this includes, among other things, the words we use, the navigation we choose, and the layout of a Web site as a whole as well as its individual pages.

Reading Norman's book may make us chuckle at how poorly designed some things are but it can cause us to pause and reflect on the usability of anything we design. The examples of a telephone or an electronic gadget like a VCR with pages of instructions, buttons with arbitrary order, lack of natural relationships, and confusing oversimplifying diagrams are easy for one to relate to. In the world of the World Wide Web we find pages and sites with similar issues when we have no idea of where to go and what to do or cannot even figure out what is the point of the page or site.

So just how do things become unusable? If one applies Norman's list of what makes things unusable, we create an environment where the necessary actions are invisible by using non-obvious commands, acronyms and techno jargon. Messages are uninformative and actions are illogical. And be sure to be inconsistent; pages should be totally different from one another and not conform to accepted standards. Even though we feel we would never do these things unintentionally, we might. Instead, we should strive to follow Jacob Nielsen's usability heuristics which stress visibility where the users are kept informed of what is going on and given appropriate feedback. Words, phrases and concepts are familiar to the user; there is natural and logical order, relevant information, consistency and standards. There is also flexibility and good messages that reduce memory load by making things recognizable rather than relying on recall.

The bottom line is that in order to make systems usable one needs to do usability testing. Without user testing and input from our users, we may have a tendency to design for our own needs and tasks that could be different from those of our users. Usability testing is concerned with testing to see if designs are intuitive, but it also needs to be concerned with understanding the community of users who will be using the system and the actual tasks they will need to accomplish. Many times we think of usability in terms of accessibility or "look and feel" but it goes far beyond that. The ISO definition refers to usability as the effectiveness, efficiency, and satisfaction in the context of realistic tasks that are performed by the intended users; in other words, systems need to be designed that are easy to use, efficient, and pleasant to use. Internal and external consistency is important so that users can learn something once and apply it again not only in the current environment but in other environments as well.

Usability must be thought of in the context of the actual tasks that something will be used for. This will be different in different settings; as an academic setting may be different than a corporate one. There can also be different desirable levels of usability especially in learning environments where a less usable system may be acceptable because it is the user's job to learn. What is important is that usability testing analyzes the intended users and their tasks; the design draws on what the intended users know, makes sense for the required tasks, and is consistent across this and other environments.

Ben Shneiderman's book *Designing the User Interface*, now in its fourth edition, uses the term usability engineering where designs are based on the careful observation of the users and where design is a dynamic process which can change at any time based on prototyping, usability and acceptance testing. There are many user testing techniques including focus groups, interviews, talk-aloud, questionnaires, observations, and card sorting. Yet there are limitations to usability testing and any design should be supplemented with other evaluation tools such as expert reviews, cognitive walkthrough, heuristic evaluation, and surveys. These all focus on an evolutionary development that goes on during the whole design process. Even after system release continued evaluation and revision is essential.

Usability engineering for library web pages, is this overkill? How did we get here from the simple desire to create systems that are usable? Though all this testing and retesting may seem a bit daunting, any usability testing is better than none. Some simple techniques will yield a wealth of information that help us focus on what is understandable for our users, what is relevant for our users, and the tasks that need to be accomplished. When we think of our community of users and how our web sites are used, we can see that the impact of unusable designs can be non-use, the use of competitive environments such as Google, frustration, and negative perceptions of our online library resources. Whether it is called usability, user-centered design, or usability engineering, current learning theory is stressing the need to design environments that help users focus their cognitive resources on the learning task. Keeping usability and usability testing in the forefront of our designs and creating an environment that is continually being evaluated will help our users to use these environments and allow them to focus their attention on the tasks at hand. This will ultimately help our users -- that is what usability and user-centered design are all about.

Bibliography

Nielsen, J. Ten Usability Heuristics. Retrieved July 25, 2005, from the World Wide Web: http://www.useit.com/papers/heuristic/heuristic_list.html.

Norman, D. (2002). The Design of Everyday Things. Basic Books: New York.

Shneiderman, B. (2004). Designing the User Interface: Strategies for Effective Human-Computer Interactions. Addison Wesley: Boston.

Chapter 7. Surveys

Tom Lehman, University of Notre Dame

Intro

To successfully implement digital library collections and services, you'll need to get to know your users: their attitudes, behaviors, wishes, needs and desires. The more you know about them, the better you'll be able to provide services and collections that effectively meet their needs.

There are several ways to gather information about your users, including one-on-one conversations, focus group interviews (described elsewhere in this manual and surveys. This chapter provides a basic overview of surveys.

Why do a survey?

In general, surveys are carried out to gather information needed to support decision making. There are many reasons a library might want to conduct a survey:

- Determine the demand for a planned product or service
- Improve an existing service
- Find how users rate the library's current products or services
- Identify unmet user needs
- Discover problems with existing products or services
- Use library resources more effectively
- Gather input for strategic planning

Surveys compared to focus group interviews

Surveys, because they gather comparable information from relatively large numbers of people, are good for gathering quantitative information. They can tell you about characteristics of your users: demographics, how often they use library services, their attitudes towards library services. They are less useful at answering "why" sorts of questions; for example, why your users like or dislike a particular library service. Focus group interviews are better at gathering qualitative information, as they allow follow up questions and probing into responses.

Steps in Survey Design

Effective survey design involves the following steps:

1. Articulate survey goals
2. Decide who to survey
3. Decide how to administer the survey
4. Develop the survey questions
5. Pretest the survey
6. Conduct the survey
7. Analyze the results and document the findings

We'll discuss each of these in turn.

Articulate survey goals

This is an important step, frequently overlooked. Drafting and writing down a concise statement of the survey goals will be invaluable as you proceed with the process. It will help you as you draft your questions, select who you will survey, and decide how you will conduct the survey and analyze and use the results.

Decide who to survey

The group of people you want information on is known as the population of interest, or target population. In a library, that might be the users of the library's services, or those the library is meant to serve. In other contexts, it might be a smaller group, such as employees of an organization. If the population of interest is small enough and the members known, every member of the target population can be surveyed. This is known as a census survey. In cases where the target population is to large for a census survey to be practical, or the identity of some of the members is unknown, only a portion, or sample, can be surveyed.

If the sample is properly selected, survey results can fairly accurately represent the responses you would have gotten from the target population as a whole.

There are two issues to address with survey samples: sample selection and sample size.

Sample selection:

There are several ways to select a survey sample:

a. Random sample: A random sample is one where the probability of a given member of the target population being surveyed is equal to that of every other member, ensuring the people surveyed are representative of the target population. If those selected to be in the sample are not representative of the population as a whole, bias can occur and the results may be inaccurate. [1]
b. Systematic sample: A systematic sample, selects every nth person in a list to be surveyed. This technique can approach random results unless the list is clustered or structured in some way.
c. Stratified sample: A stratified sample is used when the target population is composed of several subgroups with differing characteristics, such as faculty and students, to ensure an adequate number of responses from each group. Stratified sampling selects from each of the subgroups a number of individuals to be surveyed, in proportion to their numbers in the target population.
d. Convenience sample: A convenience sample is one that surveys whoever is easily available, for example, by putting a link to the survey on the library website, or handing out surveys to persons present in the library. This type of survey can provide useful information, but the results should not be viewed as being representative or statistically valid, as there is no way of knowing how closely those who took the survey resemble the target population.

Sample size:

How many people should one survey? Alreck and Settle (1985) [2] recommend a minimum of 30 and a maximum of 10% or 1,000, whichever is less. Factors indicating the need for a larger sample include: 1) The decision that needs to be made is important and costly; 2) The target population is diverse. Conversely, if the decision is less important, the population is homogeneous, or only rough estimates are needed, a smaller sample size may be adequate.

It is true that the greater the number of responses, the more useful information one is likely to get, particularly if the survey has a section for comments. However, going from 250 responses to 1,000 responses will only double the validity of the results. [3]

Methods for Administering Surveys

There are several ways surveys can be conducted.

- One on one: An interviewer conducts the survey, either in person or on the phone. Advantages are that one is more likely to get an answer to every question (people are less likely to skip questions) and the interviewer can explain questions that aren't clear to the person being surveyed. A disadvantage is that people may not answer sensitive questions truthfully, or give answers they think will please the interviewer or that are more socially acceptable. Also, one on one surveys are time consuming to conduct, and to compile the results.
- By mail or e-mail: Advantages of mail surveys are that people are more likely to answer sensitive questions truthfully, particularly if the survey is confidential, and that they can be conducted more quickly. Disadvantages include the inability to clarify questions, and the fact that mail surveys typically have low response rates.
- Internet: Internet surveys have a number of advantages: they are fast and easy to conduct. People are more likely to answer sensitive questions truthfully (as long as they believe their answers will be kept confidential). People tend to provide lengthier comments on Internet surveys than in mail surveys. Internet surveys allow linking to a page or service one is asking about. Software is available that makes creating Internet surveys easy, and compiles the results automatically. A disadvantage is that using Internet surveys could introduce bias in that the persons taking the survey will be computer users and may not be representative of the target population.

Developing Survey Questions

Questions are the most important part of a survey. To get the right information you need to ask the right questions. Good survey questions have three attributes: focus, brevity, and clarity.

Every question should focus on a single issue or topic and should be designed to gather a single piece of information. Questions should be as brief as possible. Long questions are harder for respondents to process, and more likely to be understand. Questions should be as clear as it is possible to make them.

Questions should be reviewed by persons other than the ones who drafted them to see if there is any way to make them clearer or briefer.

General guidelines

Keep the survey as brief as possible. Aim for a survey that takes no more than 10-15 minutes to complete. You will lose people with long surveys - they may start skipping questions or answer them without really thinking through their answer.

Avoid jargon: ILS, database, serial. Unless you're doing person to person surveys, there won't be an opportunity to clarify questions. A survey respondent who answers without understanding the question won't provide usable information.

Begin with an introduction stating the purpose of the survey, what you hope to learn from it, and how the information gathered will be used. Ask those being surveyed for their help and thank them.

Questions should move from general to specific, and from easy to hard. Group questions in sections, with transitions.

Provide contact information at the end. This allows survey respondents to clarify questions they might have, or report problems with the survey.

Types of questions:

There are two basic types of survey questions: structured and unstructured. Structured questions set the format of the response, such as Yes or No, multiple choice, or asking for a number ("*How many years have you worked in the library?*").

Unstructured questions are useful where responses would be difficult to categorize (e.g. "*If you could change one thing about the library, what would it be?" "Is there anything else you'd like to tell us?*").

Most survey questions should be structured, for several reasons:

- the information they gather can be more easily analyzed, as responses are comparable
- they make the survey taker's task easier by showing the range of responses being sought
- responses to unstructured questions must be grouped and categorized, requiring analysis and interpretation

Structured questions fall into several categories:

A list of categories:

Examples:

"What is your status? [ ] Student; [ ] Faculty; [ ] Staff"
"Do you have Internet access to the library at home? [ ]Yes; [ ] No"

Rating:

Examples:

"How useful is the library's 'Ask a Librarian' service to you? [ ] Not useful at all; [ ] Minimally useful; [ ] Moderately useful; [ ] Very useful"
"On a scale of 1 to 5, with 1 being "Don't use" and 5 being "Use regularly," rate how much you use the following library services: [ ] ILL; [ ] Ask-a-Librarian; [ ] WorldCat; [ ] The library catalog"

Agreement:

Example:

	Strongly disagree	**Disagree**	**Agree**	**Strongly agree**
Library collections in my area of study are adequate	[ ]	[ ]	[ ]	[ ]
I prefer to access library materials online rather than in print	[ ]	[ ]	[ ]	[ ]
The number of hours the library is open meets my needs	[ ]	[ ]	[ ]	[ ]

The above example illustrates a question presented in a matrix format. These present information compactly, which can be helpful when used sparingly, but can also be difficult for survey respondents in large doses.

Issues related to survey question:

Responses provided to questions should be mutually exclusive.

Responses provided should cover all possible cases. Questions that don't allow persons to give exact responses will frustrate the respondent. One way to cover all possible cases is to provide a "Don't know" or "No opinion" option.

Demographic questions should be put questions at the end, unless you are only looking for particular type of respondent. They are more likely to be answered at the end.

If doing an Internet survey, allow adequate space for comments. A web survey form with inadequate space for comments will frustrate respondents whose comments are cut short, and prevent you from getting useful information. Unsolicited comments on surveys can be golden.

Some articles on surveys advocate giving an even number of choices. The theory is that with an odd number of choices, people are more likely to choose the middle option (e.g. 3 on a 1-5 scale).

One source of survey error is habituation. If the survey has a series of question whose answers are structured identically, users may select the same option for multiple questions. One way to address this is to change the answer format or direction of the scale (1 is best vs 1 is worst). This will make it more likely respondents will address the questions independently.

Pretest the survey

Pretest the survey with member of survey's target population. Testing will likely reveal potential problems that would keep the survey from yielding useful information, such as unclear questions or instructions.

If doing a Web survey, test it on a variety of browsers.

Conducting the survey

If you are surveying a predetermined sample via mail, e-mail or Internet, you will want to send out prenotification letters to let the people know the survey is coming. The letter should ask the individual to help by taking the survey, explain the reason for the survey, when it is being done, who is sponsoring the survey, what the benefit will be, and the confidentiality policy (if applicable). Research shows that letters coming from the top person in the organization, e.g. library director, result in higher participation rates.

Several days after the survey has been delivered, a letter thanking the survey participants and reminding them to complete and return the survey should be sent out.

A week or so any after the survey was sent out, any non-respondents should be sent a follow-up letter reminding them to complete the survey.

Analyzing survey results

In analyzing survey results it is useful to describe the central tendency for numeric data. To do so, the terms mean, median, and mode will be needed.

Mean (also known as "average") is the total of the values divided by the number of values. If Joe scored 87% on a test, Marty scored 95% and Fred scored 72%, the mean test score would be 254 (total of the scores), divided by 3 (number of values), or roughly 86.7%.

Sometimes the mean can be misleading, and other measures of the central tendency are better. If the incomes of the top five officers of a company were:

- $2,500,000
- $137,000
- $107,000
- $88,000
- $74,000

the mean salary would be $581,200. A more useful statistic in this case is the median salary. The median is the middle case, where half are above and half below. Here the median salary would be $107,000.

To describe the central tendency for category data, the mode is sometimes used. The mode is the category with the highest total. If the enrollment at a university is as follows:

College of Arts and Letters:	2,874
College of Business Administration:	1,595
College of Science:	953

the mode would indicate that a randomly selected student would be most likely to be enrolled in the College of Arts and Letters.

The most commonly-used tools for analyzing survey results are frequency tables and cross-tabulation tables.

A simple example showing analysis of category data using a frequency table:

Category	**Frequency**	**Percent**
Undergraduate student	87	77.0%
Graduate student	13	11.5%
Faculty	9	8.0%
Other	4	3.5%
Total	113	100.0%

Cross-tabulation tables is another commonly used technique showing the relationship of two categories.

```
+ ----------------+-----------+--------------+---------+---------- +
|     College     | Undergrads | Grad students | Faculty | Row total |
+ --------------- + ---------- + ------------- + ------- + --------- +
| Arts and Letters |    29     |       7       |    5    |    41     |
|   Row %         |    70.7%   |     17.0%     |  12.1%  |   100%    |
|   Col %         |    33.3%   |     53.8%     |  55.5%  |    37.6%  |
+ --------------- + ---------- + ------------- + ------- + --------- +
|     Business    |    38      |       2       |    2    |    42     |
|   Row %         |    90.4%   |      4.7%     |   4.7%  |   100%    |
|   Col %         |    43.6%   |     15.3%     |  22.2%  |    38.5%  |
+ --------------- + ---------- + ------------- + ------- + --------- +
|     Science     |    20      |       4       |    2    |    26     |
|   Row %         |    76.9%   |     15.3%     |   7.6%  |   100%    |
|   Col %         |    22.9%   |     30.7%     |  22.2%  |    23.8%  |
+ --------------- + ---------- + ------------- + ------- + --------- +
|      Total      |    87      |      13       |    9    |   109     |
|   Row %         |    79.8%   |     11.9%     |   8.2%  |   100%    |
|   Col %         |   100%     |    100%       |  100%   |   100%    |
+ --------------- + ---------- + ------------- + ------- + --------- +
```

Other statistical tools are available for more complex analyses of survey results consisting of large sets of continuous numeric data. If interested in more complex analyses of survey data, consult those with expertise in this area or advanced textbooks.

References:

[1] A well-know example of survey bias is that of the *Literary Digest*. Prior to the 1936 presidential election it mailed out millions of postcards asking people's preference. "Its 2.3 million "voters" constituted a huge sample; however they were generally more affluent Americans who tended to have Republican sympathies. The *Literary Digest* did nothing to correct this bias. The week before election day, it reported that Alf Landon was far more popular than Franklin D. Roosevelt. At the same time, George Gallup conducted a far smaller, but more scientifically-based survey, in which he polled a demographically representative sample. Gallup correctly predicted Roosevelt's landslide victory." (http://en.wikipedia.org/wiki/Opinion_poll)

[2] The Survey Research Handbook, by P. Alreck and R. Settle. R.D. Irwin, 1985. 400 p.

[3] The Survey System: http://www.surveysystem.com/sdesign.htm

[4] Survival Statistics: Designing and Using Surveys, by David S. Walonick, c1997-2004 http://www.statpac.com/surveys/surveys.doc

Chapter 8. Focus Group Interviews

Megan Johnson, Appalachian State University

What is a Focus Group?

Focus groups are discussions that last one to two hours, usually done with 6-12 people. The discussion is led by a moderator asking open-ended questions. In usability research for web sites, focus groups are used to explore perceptions and gather opinions of your interface and patron needs.

Why are They Helpful?

Focus groups can be used to discover people's general reactions to an interface or services. A wide range of information can be gathered in a relatively short time span, and it is a good device to find out what users want from your site. For example, if you are testing your web site with current users, and they all express the concern that the interlibrary loan feature is very difficult to navigate, that is concrete information.

Focus groups are particularly helpful when used in conjunction with surveys. Your library can follow up a survey with focus groups to clarify the issues revealed, and perhaps hear surprising new ideas or concerns. Unlike doing polls or asking a listserv, the strength of this technique is the interaction between participants. With a skilled moderator, the conversation can go beyond "like it, don't like it" and allow new views to surface. Since there is no pressure to reach a consensus, all views can be encouraged and aired.

The "con" side of focus groups is the results cannot be generalized or treated statistically. A focus group analysis is qualitative. Jacob Nielsen cautions that "Since there are often major differences between what people say and what they do, direct observation of one user at a time always needs to be done to supplement focus groups." Some believe that focus groups are not a good way to gather information. Malcolm Gladwell, author of Blink, is quoted in Advertising Age January 24, 2005, "There is very little psychological justification for the notion that you can find out what people think about an idea -- particularly a revolutionary new idea -- by asking them."

However, Microsoft and other high tech companies use focus groups, but they don't just pick random users. They carefully choose the membership of the groups. They find advanced customers by checking message boards and Web sites to see who is answering technical questions about their products. High tech companies seek input from advanced customers to improve current products and products in development, and find their input valuable.

To apply this to a library setting, you may want invite different constituencies to separate focus groups, for example, asking graduate students writing their thesis as one group and computer science majors as another, and library staff as a third. However, recruiting participants is often difficult, involving lots of soliciting and following up with calls and e-mails.

How to Conduct a Focus Group

To prepare for a focus group, the steps are to make sure you understand what information you are trying to gather, develop relevant questions, schedule and facilitate the meeting, and then analyze the results. Details of these steps follow.

The first step is to understand what you want to learn. Focus groups are not polls or surveys; they are in-depth, qualitative interviews with a small number of carefully selected people, to help you develop an idea or specific service.

To prepare, you may want to first consider your budget. This is a labor intensive project; time may be the most expensive item. It takes considerable time to plan the sessions, recruit volunteers, and develop your questions. If you hire a professional moderator, that could be a considerable expense, other optional things can add to the cost, including honorariums, refreshments and video taping.

Much of the literature refers to a "trained moderator" leading the discussion. Unless you hire an independent consultant to do this, you need to build the skills to run a focus group or ask an experienced neutral colleague. Practice will help improve skills, so build in a "test" focus group. The moderator cannot be too close to the project under discussion. If it is a general library focus group, it is probably best to not have a person who works at the library conduct the session, since the participants may not want to offend, and the results from a librarian could have bias.

What skills are needed? A skilled moderator has excellent interpersonal skills and keeps the focus! They keep the discussion on track so that none of the material that is intended to be covered is omitted, and they encourage discussion without influencing the answers given. A good moderator ensures that each of the people in the group participates and interacts with the others and that the discussion is not dominated by one or two individuals.

They remain neutral, and encourage responses with occasional nods of the head, and phrases like "uh huh". The moderator steers the discussion, using phrases like, "we've been talking about desktop delivery, and now I'd like to move on to the format of the documents."

The moderator keeps control of the interview and maintains momentum. If the conversation gets off topic, he or she can use phrases like, "that is an interesting point, but let's stay focused on..." If someone is not participating, the moderator can directly ask their opinion.

The next step is membership. Who do you invite to participate? Again, this depends on what you are trying to learn. When developing new features, some gaming companies recruit avid players. Whether you are thinking about offering a new service or building a new building, your must carefully choose your membership, and perhaps plan several groups. If you are gathering data on your library web site, you may want to plan several groups -- one for staff, another for freshman, one for graduate students, and one for teaching faculty. You can plan a practice session with librarians or student workers to help you refine your strategies.

When you choose your target members, you can solicit them in any way that works locally -- send them and e-mail or give them a call, and explain the time commitment and if there is any remuneration. Speaking of this, should you give an honorarium to participants? This it depends on your budget and your situation. You may consider a voucher for the bookstore or ten to fifteen dollars an hour, or you could hold the session over lunch and provide a meal.

The next step is developing your questions. Experts recommend using only about six or seven questions in a focus group, so this step takes some planning. The questions will center on what you need in-depth information about. The questions will be different if you trying to learn if a new service or idea will work, or if you want to understand where a web site is confusing.

Guidelines for the questions include: make questions as neutral as possible, that is, don't use disparaging language or describe something in enthusiastic terms. For example, if you are discussing a service, just name the service and describe it if needed, without using terms like, "popular" or "costly". Make the questions open-ended (i.e. if it can be answered with a yes or no, don't use it). When listening to answers, don't allow one participant to finish another's answer.

Avoid "why" questions. The answer to a "why" question may conclude a false cause-effect relationship. Also, be careful asking "why" in following up with a participant, in case the respondent feels challenged or defensive. If an answer surprises the moderator, instead of asking "Why do you think..." a phrase like, "Can you tell me more about that?" might get a more nuanced response.

Plan the sequence of questions. To get everyone involved, first ask some factual questions, before asking for opinions. For the last question, ask respondents if they have any other comments (try a round robin for this) and if they have any feedback on the process of the focus group itself.

Planning the Session

Set the mood: a focus group is an "event" for your participants. Hold sessions in a pleasant room, free of distractions. Configure chairs in a circle or around a table so that all members can see each other. Provide name tags and refreshments.

To facilitate, after introducing yourself and the observer (if used) first review the purpose of the meeting, and how the results will be used. Address terms of confidentiality, who will get access to their answers and how their answers will be analyzed. If you will be using direct quotes, or if your organization requires consent forms for human testing, have participants sign the necessary forms.

Tell the group how long the meeting will last. Most focus groups run one to one to two hours long (overestimate by 15 minutes and do not run over). Go over ground rules, such as, we are here to gather opinions, let everyone participate, and if participants will receive remuneration how that process will work. Explain the means to record the session (note taking, audio or video recorder). Ask participants if they have any questions before getting started with the interview.

At the end of the session, thank the members, remind them their input was helpful, and restate how the day's results will be used. Tell them how to get in touch with you later if they have further thoughts.

Should the event be recorded? It adds cost and complexity to use a video camera, but it gives others of the team a chance to view the results later without sitting in the room, or behind a mirror (possibly influencing the participants). It can also help the moderator build their skills ...did they raise their eyebrows, or frown at a comment? If you do not record the event, it is important to have an non-participating observer in the room (or behind a mirror) unobtrusively take notes.

After the session, write a short summary, grouping results into specific ideas. Include anything that was "not what you expected" and relevant quotes to demonstrate concepts.

What about alternatives such as using electronic discussion lists or group chat for a version of a focus group? You can gather very valid input this way, but it does not include the advantages of being able to observe non-verbal reactions and you can lose or change the group dynamic. Also, the confidential nature of a focus group is lost. Jacob Nielson says, "Posting questions to a newsgroup with an interest in the issues can generate considerable discussion. A disadvantage is that online discussions are difficult (or impossible) to keep confidential unless they take place on an intranet, behind a firewall. Although online forum discussions are unlikely to reflect the average user's concerns, they can be a good way of getting in touch with "power users." These users have needs that will sometimes surface later for the average user. Thus, addressing the power users' needs may be a way of getting a head start on future usability work."

Sources and Helpful Web Sites

Basics of Conducting Focus Groups by Carter McNamara, Ph.D. http://www.mapnp.org/library/evaluatn/focusgrp.htm

Focus Groups - Articles by Tom Greenbaum of Groups Plus, Inc. http://www.groupsplus.com/pages/articles.htm.

Six Sigma: Focus Groups - http://www.isixsigma.com/vc/focus_groups/

Social Research Update 19: Focus Groups. Sociology at Surrey. University of Surrey. http://www.soc.surrey.ac.uk/sru/SRU19.html

The Use and Misuse of Focus Groups by Jakob Nielsen - http://www.useit.com/papers/focusgroups.html

What Are Focus Groups? American Statistical Association. Pamphlet. 8 pages. http://www.amstat.org/sections/srms/brochures/focusgroups.pdf

Chapter 9. Attracting Users

Michael Yunkin, University of Nevada, Las Vegas Libraries

Introduction

Usability testing can't begin without test subjects, but recruiting them often takes a lot of work and creativity. In an academic setting, it would seem easy to find volunteers; after all, we deal with students every day. In practice however, it's just not that easy. Students are busy, often difficult to contact, and not always reliable. Successful recruitment too often means 'making it worth their while' -- not an easy thing to do on a limited (or non-existent) budget.

Fortunately, there are a number of ways testers can attract volunteers. Some are straightforward (money!), and others more subtle. This article is a compilation of various methods academic libraries might use to recruit test subjects, based on the experience of the author.

Incentives on the cheap

If you have a budget, getting incentives is easy; there is an endless list of things students might want enough to give up an hour of their time. On the other hand, many of us aren't lucky enough to have a usability budget, and need to use a little imagination to lure students into participating.

1. **Gift Certificates & Coupons.** People who've never worked in retail might not realize it, but many retail establishments and restaurants have small budgets set aside for donations. Campus coffee shops might be willing to donate some coupons to the library, and so might the university bookstore. Moving off-campus, Borders Books and AMC Movie Theaters are generally happy to provide coupons or gift certificates, and there are probably many other businesses in your area that would love the free advertising. Don't think you're asking for a handout by hitting these businesses up for donations-there's plenty of quid for their quo. When a business donates a gift certificate for you to give away to a student, not only does their kindness result in guaranteed extra foot traffic, but smart businesses know that they will almost certainly make more money back than they donate. That's why so many are happy to give away gift certificates rather than merchandise.

2. **Look around the library.** Do you charge for printing and photocopies? If so, a coupon or copy card for free copies or printouts might be feasible. Or perhaps you can offer to forgive overdue fines. On one occasion, my library's marketing committee put on an event and had some leftover gift bags of pencils, pens, erasers, and other miscellaneous school supplies, and we were able to use even these simple items to help bring in participants. The moral of this is that you don't have to offer them much, just something for their time.

3. **The greatest gift of all: Extra Credit.** If you can get teaching faculty on board, extra credit is a practically fool-proof way to get students interested. This method might be particularly successful if you're designing a website with a particular user group in mind (like a government information site for political science majors, or a branch website).

Locating willing participants

Even with incentives, finding subjects to take your test can be a chore. Here are some recruitment ideas that might make the process a little less painless.

1. **News item on the homepage.** Advertising for subjects on the library's home page may not result in a mad rush of willing participants, but it's free, and if you stress the fact that you have some sort of incentive, you're almost guaranteed some success.

2. **Other testing methods.** It's a lot easier to get students to take an online survey than it is to get them into the library for a usability test, but surveys might be a good way to find students who are interested in helping out more. Try posting a short, simple survey on your homepage, with an option to leave a phone number or email address for those willing to help out further ("You can help us further redesign our website, and get some free stuff! Let us know if you're interested!!"). Even if none are, you'll still have the additional feedback from the survey, and it's cost you nothing.

3. **Advertise.** One thing the library has is a built-in market. Don't limit yourself to library bulletin boards. Ask students at the reference desk or in instruction sessions if they'd like to participate. Create handouts, and put advertising stickers or bookmarks on the computers, telephones, and reference desks.

4. **Student workers.** Obviously, the more 'average' a subject is, the more valuable their test results will be. On the other hand, library student workers are a captive audience. Though they may have more than the average understanding of library workings and terminology, they can still provide valuable insight into new interface designs, particularly at the early stages. What's important here is unfamiliarity with the material being tested. A student whose work requires constant use of the library online catalog will probably not be a good subject for testing a new catalog interface. But a student worker from the Preservation department, whose sole job is binding books for eight hours a week, might be a fine addition to round out an otherwise skimpy list of subjects. The key here is not to stuff the subjects list with library workers (or dining hall workers, or bookstore workers). As long as the subjects are as representative of your user group as possible, your results will be also be representative, and your time well-spent.

5. **Graduate students.** Grad students are often more eager to participate in a website redesign than undergrads. They use your online tools on a regular basis, and recognize that they have a vested interest in making the tools as effective and user-friendly as possible. Advertise in the graduate student lounge, or with teaching assistants.

6. **Branches and subject librarians.** Due to their more narrow focus, the faculty and staff at branch libraries often work more closely with students or professors than at 'main' libraries (I can tell you that we had a much easier time recruiting students for our Architecture site usability test than we did for the main library site). These students also might feel that they have greater vested interested in having a great, user-friendly website, and therefore be more likely to help out. Even if your branch libraries don't have an independent web presence, their librarians' closer relationships with library users might make subject recruitment easier.

   Similarly, subject librarians often have particular students (especially grad students) with whom they work on a regular basis. Ask your subject librarians to pass along requests for help with the usability tests; most will have at least one student that they know might be interested.

7. **The brute force approach.** This isn't fun, but it works: Go to places in your library where students tend to socialize rather than study, and just approach random students and ask them if they'd like to help you redesign the website. This approach worked particularly well in the coffee shop when we had free coffee coupons. "We're giving away three free coffee coupons to students willing to help us redesign the library website" worked well, especially when we pointed out that "that's a 15-dollar value for about a half hour of easy work!"

8. **Use your current subjects as resources.** Ask your current subjects for names of other potential victim-er-subjects. Additionally, students who volunteered for your new branch website usability test might also be willing to help out with your OPAC redesign. Keep a list of volunteers' email addresses -- you might need to re-use them in a pinch.

Conclusion

With all the additional planning that goes into usability testing, it's easy to overlook the most important -- and potentially the most time-consuming-part of your testing: recruiting test subjects. Fortunately, with a little extra work and some creativity, finding incentives and attracting willing participants is something even libraries with no budget can accomplish.

Chapter 10. Card Sorting

Terry Nikkel
Shelley McKibbon, Dalhousie University Libraries

Introduction

One of the most important elements of the library website design process is determining how information and resources will be organized on the site. It is crucial to get user feedback as early as possible to ensure that the final navigation scheme, which exposes the relationships between site resources and features, is practical and usable. Card sorting is a procedure used to develop an understanding of how users themselves perceive and categorize library collections and services, and can thus dramatically improve website taxonomies and ultimately the usability of the entire website. Card sorting is efficient, economical, and can be done quite quickly.

Description

Card sorting studies range from informal exercises to highly structured and rigorously controlled research, but the basic design for all is quite similar. Subjects, who represent user groups, are recruited and asked to sort cards bearing labels and titles corresponding to library services, collections, and titles into groupings that seem logical and appropriate to them. Subjects are then asked to create their own labels for these groups. Analysis of these groupings and labels is often highly informative, and results in relevant and useful suggestions for improved website organization and navigation.

A card sorting study begins with creation of a set of cards that represents the information contained on or proposed for a web page or site. 3X5 index cards are fine, but slips of paper cut to size serve equally well. Names of services and resources are printed on individual cards, and include things like reference assistance, document delivery, library hours and contact information, along with important collections (catalog, theses, electronic resources, etc.), plus a sampling of titles of major reference works, journals, and databases in broad areas or specific disciplines. Large websites may require as many as 150 - 200 cards, which will take a typical user about an hour to sort and label. Digital simulations of card sorting exercises do exist, and are useful in large studies, but manual sorting of real cards is preferred in small scale studies because of the ease of set up and administration. In any size of study, analysis can be greatly aided by tools such as Microsoft Excel.

Methodology

Like many studies, card sorting is as much a logistical as an analytical challenge, but one of the many appealing features of this approach to understanding user needs is that just about any size of study will yield good results. The first tasks are selecting the labels for the cards, and recruiting participants. Assuming a team is involved in conducting the study (though a team is not absolutely necessary for a small-scale exercise), these tasks can be split up and accomplished quickly. Most cards should reflect current resources and services -- if new ones are being considered for inclusion in the new or revised site, add these to test them as well. Participants are recruited in the usual ways, for example posters, class announcements, advertisements, and so on. The offer of an incentive (a small cash payment, perhaps a gift of some kind) is sometimes needed to get users involved. Participants are told that the activity is not a test, and that there are no right or wrong answers.

A card sorting session with a participant may be divided into three steps: initial sorting, grouping, and labeling. In the first stage, each participant is given a set of labeled cards and three "place mats". The mats (laminated 8.5 x 11 sheets of paper can be used) are labeled "Resources used at least once," "Resources recognized but not used," and "Resources not recognized." The subject is instructed to sort the cards into these categories, with the place mats serving to help them keep the three piles separate. After sorting the cards the participant is given blank cards and asked to add any resources he or she has used but which were not represented by one of the pre-labeled cards.

After finishing the initial sort, the participant is given a pen and asked to label the cards in their 'used at least once' pile according to frequency of usage. The frequency categories can be something like daily (use label 'd'), weekly (w), and occasionally (at least once--o). While the participant is doing this, the moderator marks the other two piles of cards as either known but not used (k) or not recognized (?). Having participants label their cards in this way will allow the team to detect any patterns in the resources that are or are not used. The pile marked with the '?', or not recognized, is set aside.

In the second step, the participant sorts the cards into what he or she considers logical groupings. All cards representing resources used by the subject must be included in this sort. Participants can be given the option to sort any or all of the 'Known but not used resources' cards; they may know, for example, that New England Journal of Medicine is an electronic journal, but have never had occasion to use it. The moderator must be careful not to comment on any particular selection; card sorting is designed to determine how the user perceives different resources (or at least the labels used to define them).

Finally, the participant is asked to label the groupings of cards they have created. Using fresh cards of a different color (to help keep category labels distinct from resource cards), the participant writes down a word or phrase that best represents the entire group of cards. The groupings themselves are necessarily completely subjective; this is indeed the point of the study and it cannot be stressed too much. Subjects should therefore feel completely free to make up whatever labels they think best describe the groups of cards. Again, the moderator should refrain from coaching or questioning, but it may be necessary at this point to seek clarification of individual labels if it is not clear what they mean.

Analysis

As each session is completed, the team members record results in an Excel spread sheet which has one worksheet for each participant. All of the resources/services used on the cards are listed in rows, and there is a column for the user-created category the item was placed in or, if applicable, either '?' for an unknown resource or 'k' for a known but never used resource that was not included in the participant sort. There is also a column for recording frequency of use as indicated by d,w,o, or k (if the resource was never used but recognized and included in the sort). After the spreadsheet is filled in for each participant the team can sort resources into their user-created categories and look for points of comparison. Are certain resources tended to be grouped together? Do different users choose similar labels for groups of similar resources? There will likely be at least some very obvious general trends that can be used in the website design.

Excel can be used to cluster user responses to whether or not resources were 'known' (k), or 'not-known' (?) to gain insight into usefulness of current labels. Not surprisingly, commonly used resources like 'Library Hours' and 'University home page' will have broad recognition, while others like 'Proquest databases' or 'Nature (Online)' may be recognized by few, if any, participants. Information like this might lead designers to avoid placing specific titles and types of resources at the same hierarchical level in the website navigation scheme. Also, clustering can confirm trends revealed in the labels participants attach to different resources. For example, in a card sorting study conducted at a medium-sized university library, over half of the 26 participants grouped various writing-related resources together (e.g. 'Footnotes', 'How to Write Essays', 'Citing Sources in Your Writing') and gave them the label, 'Writing'. It had not occurred to designers to include such a category on the top level page before, but the study revealed that it would be useful, and it has since been included as a main page link.

Card sorting is an excellent method for testing the usefulness of links and label with real users, and can help minimize library jargon and other confusing language. Indeed, card sorting can even help keep web pages current, or at the very least ensure that definitions of resources are clear and unambiguous. In the same study mentioned above, one participant categorized the card labeled 'Cell' (a major journal title in health sciences) under 'Library Services'. When asked to explain, the subject replied that she was able to user her mobile phone anywhere in the library, and thus thought that the library had some kind of enhanced signal, which was a great service!

Chapter 11. Paper Prototyping

Nora Dimmock, University of Rochester

Introduction

Paper Prototyping is an integral part of the iterative design process that allows website designers the ability to iterate rapidly and inexpensively. In paper prototype testing the user interacts with a paper mockup, or prototype, of the product as if it were real. For the purposes of this article we'll limit the scope of paper prototypes to website interfaces, but paper prototypes can also be used to test the usability of forms, applications, and even 3 dimensional objects. Testing with a paper prototype is appropriate for rapid design iteration: it will allow you to get a good picture of how users will interact with your interface without having to spend any money or time on writing code. Although it is such a fast and easy way to elicit user input many design teams still don't use it. Jakob Nielsen thinks this is because "people don't think they will get enough information from a method that is so simple and so cheap. It feels like you're cheating." (http://www.useit.com/alertbox/20030414.html) This method of usability testing does have some limitations, however: it creates an artificial user environment that can influence the way users interact with the prototype, so it requires a skilled usability tester. Also, it can sometimes be hard to sell test results to stakeholders, especially when using quick and unsophisticated prototypes. Empirical evidence exists, however, that there is little difference in results between high fidelity and low fidelity prototypes. (Virzi, et al., 236) Paper prototype testing is most useful in the earliest stages in the design process, especially in the concept phase. When testing existing web interfaces, live testing is more appropriate.

In most situations, a paper prototype is an unsophisticated sketch of the user interface, with little or no functionality. The most elaborate paper prototypes are fully designed graphic renderings of a real web interface with graphic designs of underlying pages in the website, allowing the usability tester to reveal to the test subject where they will be taken when they "click" on a link, button, or navigate in the site. The usability tester functions as both the tester and the computer, turning pages when there are multiple prototypes. Where there is just a single interface to be tested, the tester asks the subject to elaborate on their choices: "you clicked on this link because? What do you think will happen? Where will this take you?" This helps the design team determine if they have a match between the system and the users expectations, or mental models. While a quick prototype may look unsophisticated in comparison to full color mockups, it can be much more flexible in the hands of an experienced usability tester.

Test Methodology

When setting up a paper prototype test you should use the same methodology you use when doing other types of usability tests. There are 4 major steps involved: identify the user groups, define key tasks, test the test (refining if necessary), and present results. Identifying your user groups will help you define key tasks for the website or page. For example, when testing a series of search interfaces for finding scholarly articles there may be more than one intended user group involved: a basic search page may be intended for undergraduates and novice searchers, while an advanced search page may be intended for more advanced searchers like graduate students, faculty and librarians. Obviously the key tasks should reflect the level of sophistication of these users and the tasks we assume they will use these pages to perform; where a basic search interface task may be to find an article on nuclear war, an advanced task may be to find a specific citation to test for the functionality of field searching in the design. Testing the test with other usability team members can help you find flaws in your methodology or technique.

Prototype Design

Once you have determined your user tasks and are satisfied with your protocol you can work on the development of your paper prototype. A prototype can be a simple wireframe, or drawing, of your interface sketched out in black marker or pen. You can use a variety of stroke widths for emphasis if those are elements you want to test, otherwise the site can be sketched out with the same pen. You can underline links if that is a feature of your site or you can ask your user to make assumptions by telling you that it is a link when they select text or a screen element. Mathew Klee suggests you use "the 'incredibly intelligent mouse' -- a fancy way to say we let the user decide what's a link simply by following their behavior." http://www.uie.com/articles/prototyping_tips/ (Klee, 3) Buttons can be represented by sketched rectangles, areas of text that are not essential to the design can be represented by squiggly lines, and form elements, such as text boxes and drop-down menus can be sketched in with default values written in. Prepare post-it notes with drop down menu choices ahead of time so that when those elements are selected you can "open up" the menu by pasting the post-it on the page. You can also use fully rendered graphics of the interface, but these take more time to produce. Try to anticipate any possible interaction that a user may have with the prototype so that you can be prepared to make the test as authentic as possible by showing them the result of their action. You may need to make prototypes of a number of pages in your site to get accurate test results.

Test Protocol

When testing users with paper prototypes you should use the same techniques that you use with other user tests. Put your users at ease by reminding them that you are testing the interface or design and not their skill at Internet searching or computer use. Actually, the use of paper prototypes can be an advantage in this respect: users with computer anxiety will be relieved not to have to sit in front of a "live" system and there is no possibility of system failure. Try to approximate your system as much as possible to get the most from your test: Use a paper with an hourglass, working symbol or other text or graphic to represent response time when transitioning between screens if your live system has a long response time between actions. Encourage your user to use their index finger or pens to represent their mouse clicks and to describe their actions and assumptions as much as possible: "When I click this link I think I'll go to ...". Try to be as objective as possible; it can be difficult to act as the "computer", especially when users ask you direct questions. Remind them that you are the computer and ask them to continue speaking out loud so you can elicit more information about their problem. For example, in a test where the user asks you "This is searching x database, right?" you could respond; e.g. "I can't answer that now, remember, I'm the computer, but continue to speak out loud and tell me where you think you are and where you're navigating to so we can design a system that matches your expectations".

If you are testing forms you can have them write their text input directly into the forms; Xerox copies of your prototypes will ensure that you have an ample supply of fresh copies for subsequent users. Alternatively, you can have transparencies available for users to cover the prototype so that it will be "refreshed" for the next user. When users select the arrow next to a drop down box you can paste the prepared menu over the text box and have them continue to select menu items. Presenting prototypes of search results pages can be tricky: it can be impossible to predict the search terms entered by users but a prototype of the results page should represent the important elements presented in search results. The average user is usually savvy enough to make the stretch. The Neilson-Norman Group has an excellent DVD available that demonstrates many of these techniques and shows several live tests. http://www.nngroup.com/reports/prototyping/ (Paper Prototyping)

Presenting Results

Finally, present results in a way that is most meaningful to the design team. In her book Paper Prototyping author Carolyn Snyder offers some good advice: "It's important for observers to record data in a form that won't be subject to contentious debate later. It's natural for us to filter information through our own set of ideas and prejudices, but this subjectivity means you're now dealing with opinion rather than data...." (Snyder, Chapter 11) Don't report on inconclusive results; it will just diminish the effectiveness of clearly discovered flaws in the design. Additionally, don't infer user actions; ask users for clarification when they pause or hesitate so that you can validate your observations. If you videotape your tests or use usability software with video/audio presentation software components then presenting clips of the most problematic parts of the user experience is extremely effective. Since this is a rapid iteration technique brief reports are appropriate. New prototypes can be designed that address the most important usability issues and the design can be retested. Because testing with a paper prototype allows the design team to iterate rapidly, major design flaws can be caught before they are passed along to the development team, saving both time and money. This is one of the most important reasons to add paper prototype testing to your web development toolkit.

References

Klee, Mathew. "Five Paper Prototyping Tips". Online. March 1, 2000. Retrieved 6/19/2005. http://www.uie.com/articles/prototyping_tips/

Nielsen, Jakob. "Paper Prototyping: Getting User Data Before You Code". Jakob Nielsen's Alertbox, April 14, 2003. Retrieved 6/21/2005. http://www.useit.com/alertbox/20030414.html

Paper Prototyping: A How To Video. Nielsen Norman Group 2003. http://www.nngroup.com/reports/prototyping/

Snyder, Carolyn. Paper Prototyping: The Fast and Easy Way to Design and Refine User Interfaces. Morgan Kaufmann Publishers 2003. Accessed through Books 24X7 institutional access through University of Rochester Libraries http://library.books24x7.com/

Virzi, R. A., Sokolov, J. L., and Karis, D. 1996. Usability problem identification using both low- and high-fidelity prototypes. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Common Ground (Vancouver, British Columbia, Canada, April 13 - 18, 1996). M. J. Tauber, Ed. CHI '96. ACM Press, New York, NY, 236-243. DOI= http://doi.acm.org/10.1145/238386.238516

Chapter 12. Low-cost Recording of Usability Tests

Martin Courtois, Information Technology Assistance Center, Kansas State University

Before you begin any usability testing, you will need to have a method for recording users' responses. In cases such as card sorting or open-ended testing, it may be sufficient to ask users to fill out a questionnaire or have someone write down user's responses. For task-oriented testing, where a user is asked to use a web-connected computer to perform a task or locate specific information on the site, it is difficult for a human observer to accurately and thoroughly record a user's cursor movements, mouse clicks, verbal comments, etc. Video cameras are often used to record such tests, but may not produce a clear image of the screen and cursor movements. In addition, video recording requires equipment (camera, tripod) and an operator that may impede the testing process.

Since the tests will be conducted at a computer, why not use the computer to record the test and create a "video" file that can be played back and analyzed? TechSmith's Moare (www.techsmith.com) is the premier usability recording software, and does a great job in capturing users' cursor movements, mouse clicks, pages visited, verbal comments, and facial expressions. Moare also has tools to facilitate analyzing users' responses. But this capability comes at a price: about $1200 for a single user license. Not in your budget? Hyperionics' HyperCam gives excellent results when used to record usability tests and can be purchased for less than $40.

This article gives details on configuring HyperCam software, selecting and testing a microphone, using HyperCam during usability tests, and playing back and analyzing recordings. Instructions and settings are for the Windows XP operating system, but are similar for Windows 2000.

HyperCam

HyperCam captures action from your Windows screen--cursor movements, mouse clicks, pages visited--and saves it to an AVI (Audio-Video Interleaved) movie file. Sound from a microphone is also recorded and synchronized with screen actions. This means you can record all selections made by test subjects, along with moderator's questions and subjects' responses, and play them back in real time.

HyperCam is shareware produced by Hyperionics Technology and is available only for the Windows operating system. Specific system requirements are not provided, but the software performed without a problem when tested on typically-configured XP and Windows 2000 laptops. Information on downloading HyperCam is available on the Hyperionics web site (http://www.hyperionics.com) and can be registered for $39.95. HyperCam (current version 2.13) is about a 900 Kb download and 1.2 MB fully installed. FAQ, message board, e-mail, and phone support is provided.

Other recording software is available, but either costs much more, such as TechSmith's Camtasia at $299, or may be less reliable and not have the support provided by Hyperionics. HyperCam is a good compromise and provides reliable operation at low cost.

Configuring HyperCam

Once installed, launch HyperCam by selecting Start/Programs/HyperCam. The HyperCam window opens, and you'll use this screen to make most of the changes necessary to optimize HyperCam. The following sections refer to settings made under tabs in the HyperCam window: Screen Area, Hot Keys, AVI File, Sound, and Other Options. All settings are saved automatically.

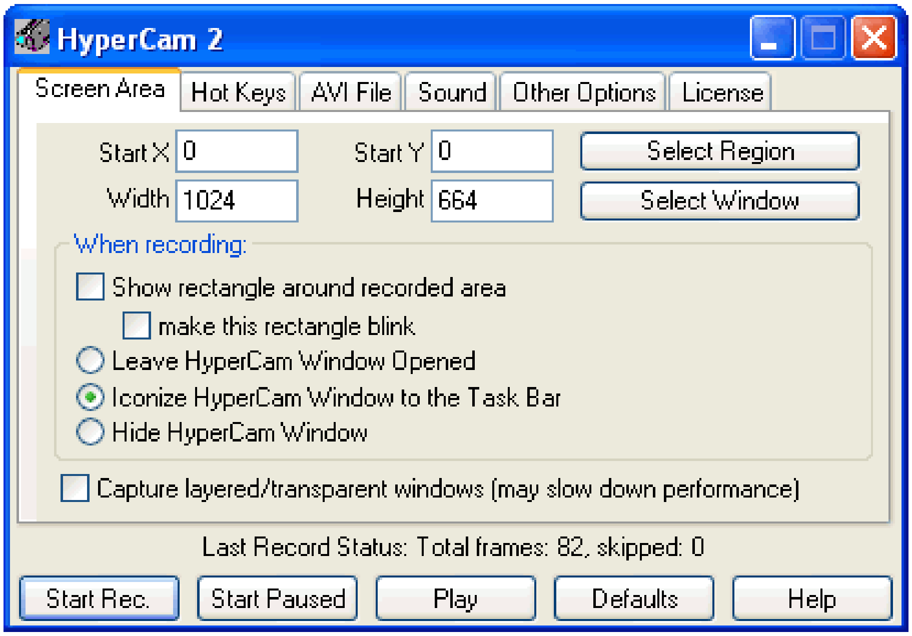
Screen Area

During testing, you'll want to make sure the recording includes the scroll bar, browser buttons, and other navigation tools. Before you set the screen capture area in HyperCam, be sure the display is set to the resolution you will be using during testing. This can be adjusted by pressing Start, then selecting Settings/Control Panel/Display/Settings. Under "Screen Resolution," move the slider to the desired resolution.

If you're happy with the resolution setting on the computer, open the browser you plan to use during testing, then open HyperCam. In the HyperCam window, click the Screen Area tab, then click the Select Region button.

The HyperCam window will disappear, and you'll see the browser window and crosshairs controlled by the mouse. Move the crosshairs to the upper left corner of the region you want to record, click, and release. Now move the crosshairs to the lower right corner and click again. The HyperCam window will reappear.

You'll want the HyperCam window to be unobtrusive during testing, so uncheck the boxes to show a blinking rectangle around the recorded area. Check the box to "Iconize HyperCam Window to the Task Bar."



Start X 0
Start Y 0
Width 1024
Height 664

When recording:

□ Show rectangle around recorded area
□ make this rectangle blink
○ Leave HyperCam Window Opened
● Iconize HyperCam Window to the Task Bar
○ Hide HyperCam Window

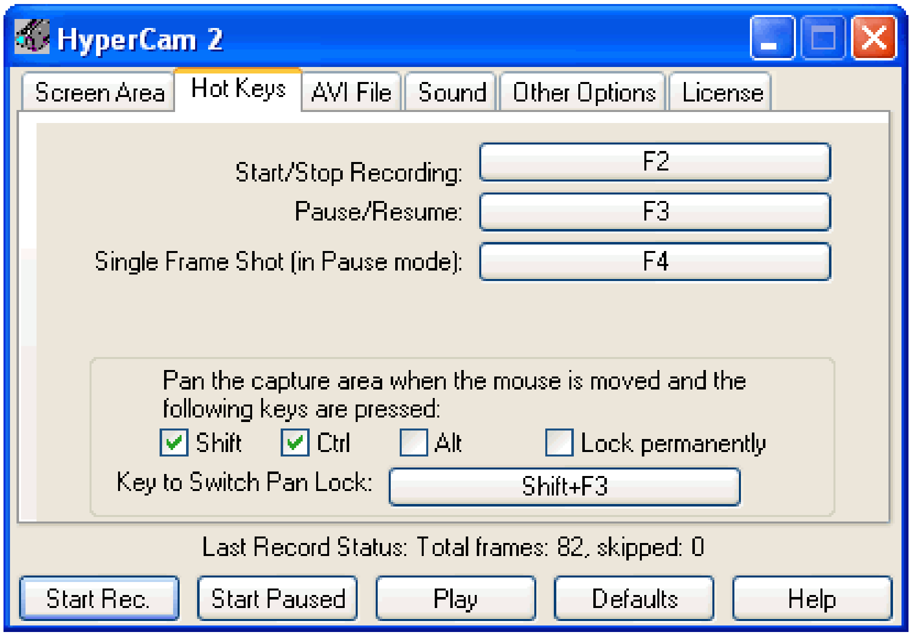
□ Capture layered/transparent windows (may slow down performance)

Last Record Status: Total frames: 82, skipped: 0

Screen Area

Hot Keys

It's useful to have a single "hot key" to start and stop recording. Default setting is F2; click the "F2" button if you need to change this. You'll want to record the entire browser window, so you won't need to use the pan function.



Start/Stop Recording: F2

Pause/Resume: F3

Single Frame Shot (in Pause mode): F4

Pan the capture area when the mouse is moved and the following keys are pressed:

☑ Shift ☑ Ctrl □ Alt □ Lock permanently

Key to Switch Pan Lock: Shift+F3

Last Record Status: Total frames: 82, skipped: 0

Hot Keys

AVI File

HyperCam can automatically generate sequential filenames (clip0001.avi, clip0002.avi, etc.) as you record. To enable this feature, enter the path and a generic filename in the "AVI File Name" box. Notice the initial, default filename is "clip0001.avi." You may want to have several directories set up for different test sessions or questions and use the "Browse" button to select the appropriate folder. Check the box "Add sequential number to the file name." With this feature, HyperCam will automatically append a sequential number to the generic file name and save the file each time HyperCam recording is stopped. This automatic process is quicker and easier than trying to type in new file names during a test session.

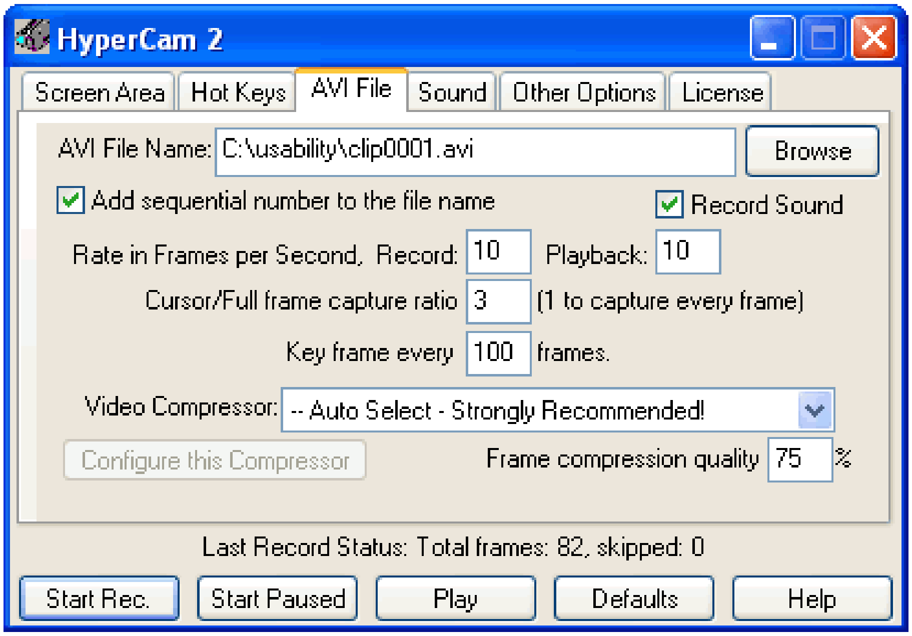
These files will be large, as much as 10 Mb for a one minute recording. To facilitate reviewing the files by different parties, it's easiest to record the sessions on a networked drive. If this isn't possible, you'll probably have to copy files to CDs or DVDs.

Check the "Record Sound" box. Additional configuration for sound will be made under the "Sound" tab.

The following changes help to keep file sizes small and will optimize performance, especially on older computers:

Cursor/Full frame capture ratio: Change to 3. This means that only every third frame will be captured in entirety. For other frames, only cursor position is updated. Since your subjects will be working on "static" web pages, it's unlikely any data will be lost. Even with this setting, full frames are captured about every .33 seconds, quick enough for even the fastest mouse clicker!

Frame compression quality: Keep at 75%. This will give crystal clear playback, although files will be large. If you need to reduce file size, change this setting to 50%. Playback will still be legible, though not as clear.



AVI File Name: C:\usability\clip0001.avi

☑ Add sequential number to the file name

☑ Record Sound

Rate in Frames per Second, Record: 10 Playback: 10

Cursor/Full frame capture ratio 3 (1 to capture every frame)

Key frame every 100 frames.

Video Compressor: -- Auto Select - Strongly Recommended!

Frame compression quality 75 %

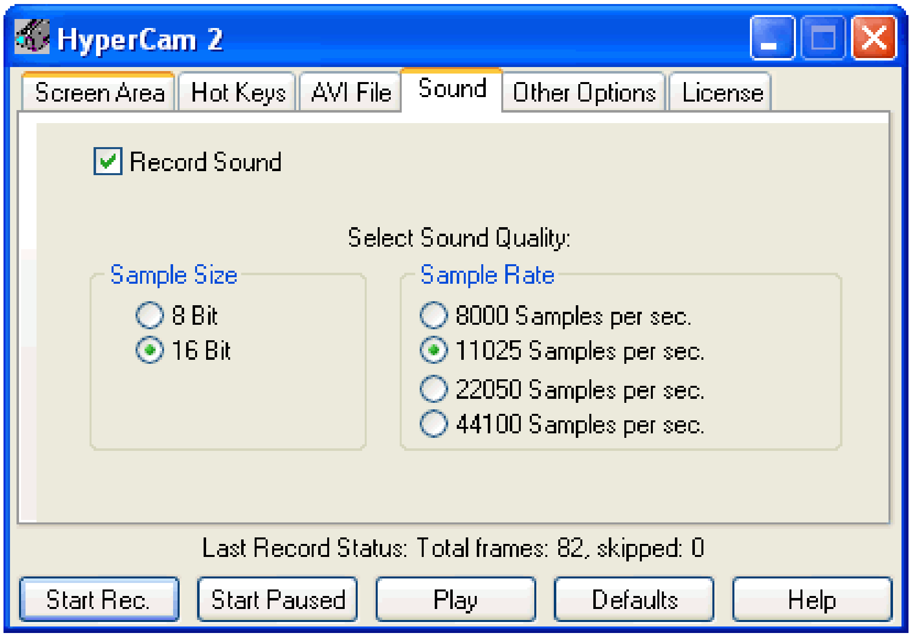
Last Record Status: Total frames: 82, skipped: 0

AVI File

Sound

Make sure the "Record Sound" box is checked.

Set "Sample Size" to 16 bit and "Sample Rate" to 11025. This gives acceptable performance for recording subjects' comments without inflating file sizes. You don't need CD-quality for these recordings, but you do need to be able to decipher what users are saying.



☑ Record Sound

Select Sound Quality:

Sample Size

○ 8 Bit
● 16 Bit

Sample Rate

○ 8000 Samples per sec.
● 11025 Samples per sec.
○ 22050 Samples per sec.
○ 44100 Samples per sec.

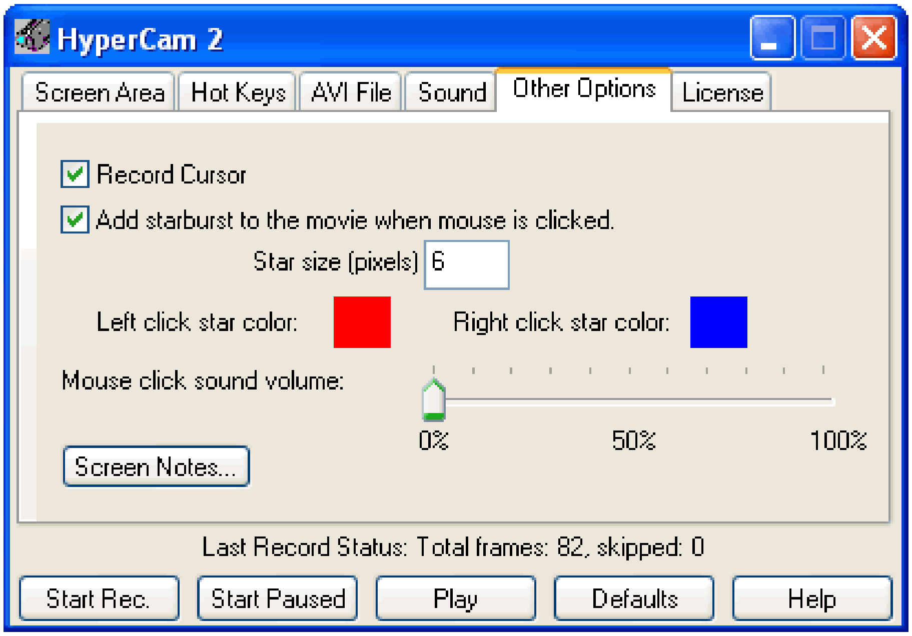
Last Record Status: Total frames: 82, skipped: 0

Sound

Other Options

Be sure "Record Cursor" box is checked.

The "starburst" feature makes it easy to determine when a subject clicks the left or right mouse button. The starburst is visible only on playback and helps to determine if the subject actually clicked on a link or only "hovered." If you want playback to include an audible click when the user makes a mouse click, move the "Mouse click sound volume" slider to the right.



☑ Record Cursor

☑ Add starburst to the movie when mouse is clicked.

Star size (pixels) 6

Left click star color: 🟥 Right click star color: 🟦

Mouse click sound volume: 0%

Last Record Status: Total frames: 82, skipped: 0

Other options

Other Settings

Color Mode

Modern computers can display millions of colors. This is a real boon for most applications, but for recording purposes, it increases file size and can slow the recording process. If color quality is not a primary concern for purposes of your testing, set your color mode as low as possible. To do this, click Start, then select Settings/Control Panel/Display. Click on the tab "Settings," and under "Color Quality," select the lowest setting. Older computers may have a selection for 256 colors (8 bit), while on newer computers, the lowest selection may be "Medium 16 bit." Be sure to restore this setting after your test session.

Hardware Acceleration

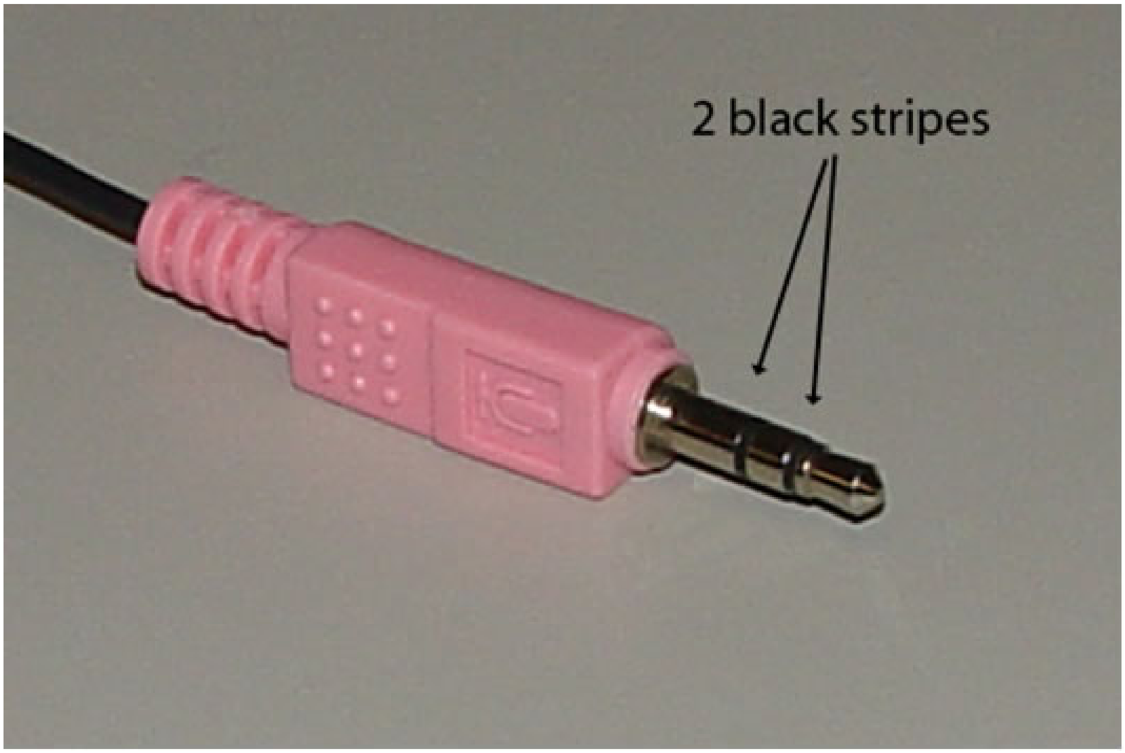
On older computers, you may notice HyperCam drops some frames and segments of the audio are lost or that the audio and video do not remain in sync. If more than 50% of frames are dropped, you will see an error message. To correct this problem, click Start and select Settings/Control Panel/Display. Click the "Settings" tab, then click the "Advanced" button. Click the "Troubleshooting" tab. Under "Hardware Acceleration," move the slider all the way to the left, toward "None." This will help to keep the graphics display in sync with the slower processor. Be sure to restore this setting after your test session.

About Microphones

Having a record of subjects' comments is a useful tool in accessing usability, but it can be tricky to obtain a legible recording. Follow these guidelines to get the best recording:

Use a cheap microphone

Sound cards on Windows computers are designed to work with 3 conductor, 3.5 mm (1/8") plugs typically found on inexpensive desktop computer microphones (see illustration). This wiring scheme is usually used to supply low-voltage DC power to the microphone, which results in a strong signal to the sound card. Notice this plug has 2 black "stripes" around the circumference of the shaft; plugs with only 1 stripe will not work as well. Professional-quality microphones use a different wiring scheme and usually don't produce a strong enough signal without an external preamplifier (1). For ease of use and a good recording, the Logitech Desktop Microphone (980240-0403) is an excellent choice and available from many online retailers for about $10.



3 conductor 3.5 mm (1/8") plug

USB-powered microphones, such as the LogiTech USB Desktop Microphone (980186-0403) are available for about $20. USB microphones contain their own analog-to-digital converters and produce a strong signal. These microphones require a slightly different setup to work properly with HyperCam, so follow the steps below based on your microphone.

Adjust the Recording Volume

Conventional Microphone (3.5mm plug)

Plug the microphone into the microphone input on your computer. This input is typically marked with an image of a microphone and is often red or pink in color.

In Windows, Click Start, then select Settings/Control Panel/Sound and Audio Devices, then click the "Audio" tab. Under "Sound Recording", click the "Volume Control" button. On the Recording Control window, make sure the "Select" box in the Microphone channel is checked. Set the volume slider for

Microphone at about 1/2. Click the Advanced button. Make the sure the "MIC Boost" box is checked. If your computer does not have the MIC Boost option, you'll need to set the volume slider higher. Click Close.

USB Microphone

Plug the microphone into a USB port on your computer. Turn on the mic (the LogiTech mic has an on/off switch on the base).

In Windows, Click Start, then select Settings/Control Panel/Sound and Audio Devices, then click the "Audio" tab. Under "Sound Recording", pull down the "Default Device" window and select the microphone. With the LogiTech mic, the selection is "AK5370," which is the analog-to-digital converter chipset used in the mic. Click "Apply." Under "Sound Recording," click the "Volume" button. This will open a Volume Control for "Wave In." Adjust the volume slider all the way up and make sure the "Mute All" box is unchecked.

Test the Mic

Click Start, then select Programs/Accessories/Entertainment/Sound Recorder. Click the red button to begin recording, and speak into the microphone. You should see the green line react as you speak. If the green line thickens only a little bit as you speak, you need to raise the microphone volume slider in the Volume Control window or move the mic closer to you. If the green line grows to fill the entire window, you need to lower the volume. Use the Sound Recorder controls to stop, rewind, and play the recording. Use the Volume Control window to adjust the playback volume. If you can get a good recording level with Sound Recorder, you will get similar results with HyperCam.

Your Sound Recorder recording is likely made at the default, 8-bit sample size, which means you'll hear some static during playback. In HyperCam, we've changed the sample size to 16 bit, so playback will be much clearer.

As you record with HyperCam, you'll need to turn off the speakers or use the Volume Control to turn down the playback volume in order to avoid feedback (a high pitched, squealing sound coming from your computers' speakers). If the moderator and subject are seated at the computer during the testing, the top of the monitor is usually a good location for the mic. If you're using a laptop, position the mic so subjects are speaking into it as they look at the screen.

Using HyperCam

With both the sound and video elements, there is a lot to configure within Hypercam. Be sure to thoroughly test the system prior to conducting actual usability testing sessions.

During the test, it's likely you'll have several, specific questions to pose to each subject. Since the .avi files generated by HyperCam will be large, create one file for each question. This will facilitate handling files if you need to copy them to CD or want to compare responses to the same question from different subjects. Plan ahead in terms of creating a directory structure and file naming scheme, particularly if more than one moderator is conducting tests.

When you're ready to begin recording, follow these steps:

- Open the browser window.
- Open HyperCam.
- Minimize HyperCam.

You should now see the HyperCam icon at the bottom of the screen. To begin recording, press F2 (or the Start/Stop "hot key" you identified in the HyperCam configuration). You will see the camera "lens" in the HyperCam icon change from blue to red. Press F2 again to stop recording, and the camera lens changes back to blue. This allows you to tell at a glance if HyperCam is in recording mode without having to maximize the HyperCam icon.

By using F2 to start and stop recording, it's easy for the moderator to control the creation of a new .avi file for each question. Simply press F2 when you want to begin recording, and press it again to stop the recording. HypeCam will automatically append a sequential number to the file name and save the file, so you won't need to interrupt the test with saving files, entering file names, or other tasks.

In addition to starting and stopping the recording, the test moderator will need to elicit comments from the subject, to get the person to "think out loud." This sounds easy, but it can be one of the most challenging aspects of conducting the test and getting data you can use. The moderator should have at the ready questions he/she can pose to the subject as the situation demands, such as these examples:

- Why did you click on that link?
- What did you expect to find on this page?
- What would have made that information easier to find?
- You look puzzled....tell me what you're thinking.

It also helps to try to put the subject at ease. Emphasize that you're testing the web site, not the subject's skill in finding information. Chat with the subject for a few minutes before the test begins, offer some refreshment, anything to make them feel comfortable. Even with these efforts, subjects may "freeze up" during the test, so be ready with questions to help draw out their comments and suggestions. Despite your best attempts, some subjects simply may not have a lot of verbal comments. Don't add to their stress by asking the same questions over and over if you're simply not getting a response. It's likely a "quiet" subject will be balanced by one who has a lot to say! Try role playing a test session to give moderators a chance to practice interacting with subjects and drawing out comments.

Playback

When you play back .avi files created with HyperCam, it's likely the computer you're using will open the files with Window's Media Player, and it's likely the image will be blurry. The problem is that Media Player tries to scale the video to a smaller size, but the video must be played at 100% in order to produce a clear image. Media Player versions 7, 8, and 9 have this problem, and attempts to adjust the application to play files at 100% are not always successful.

One solution is to run Media Player 9 in version 6.4 mode. To do this, locate and run the file mplayer2.exe in the Program Files/Windows Media Player folder. Media Player 6.4 will play .avi files at 100%.

A better solution is to download and install the free Camtasia Player from TechSmith at http://www.techsmith.com/products/studio/player.asp. Camtasia Player always plays back videos at 100% and gives a clearer image than Media Player 6.4.

With either solution, be sure the resolution on the playback computer is set to the same size or larger than the recording computer. For example, if the computer you used to record is set for 1024x768 resolution, the playback computer should be set for the same resolution or higher, e.g., 1280x1024.

Analyzing Results

After a testing session, most moderators come away with distinct impressions of where users were having problems. It's great to have that insight, but don't rely only on those first impressions. Take the time to watch all the videos you've created. It's useful to juggle playback order, e.g., view all responses to a particular question rather than all the responses from a particular user.

It may help to watch the videos in a group setting and stop playback for discussion. This is particularly useful in terms of identifying possible solutions to problem areas on the site. Finally, try to get librarians who may not be directly involved in testing, particularly department heads and administrators, to view at least a portion of the videos. Anyone who's been involved in testing knows how eye-opening it is the first time you witness someone using a site in a manner completely different than how you envisioned it would be used. Share that experience with those who may help to provide additional resources for development and testing.

References

1. Interfacing professional microphones to computer sound cards. Available: http://www.shure.com/support/technotes/app-soundcard.html [January 10, 2006].

Chapter 13. Communicating Usability Results

Brenda Reeb, University of Rochester

Introduction

Communication is a key component of successful usability work. Usability practitioners who plan how they will communicate test findings have more success incorporating findings into design. This section discusses several ways to effectively convey test findings and suggests several options for teaching library staff about usability methods. Plan a communication program in the beginning and the design process will proceed smoothly. Novice usability practitioners commonly underestimate the amount of communication needed.

Obviously, usability practitioners must convey test findings effectively to the web designers and other stakeholders who are not direct participants in the testing. If the usability practitioners also act as the web designers, communication is reduced. The usability practitioners can choose from several methods to communicate findings, or results. Verbal, written or video formats are the most common methods used in library usability programs. Each choice has trade-offs in terms of time investment and formality.

Findings: observations, opinions, design recommendations

Before discussing methods, a few words about usability findings. What is a finding? Usability practitioners must carefully distinguish among *observation*, *opinion*, and *design recommendation* when discussing usability results.

Here is an example of an *observation*, recorded in response to a test question asking the user to "Find an article on film noir for your English writing class" --

She typed the keywords "film noir" into the library catalog search box

Several possible *opinions* of this action might be --

- She knows what the catalog is and believes articles are found there
- She doesn't know what the catalog is and believes she is searching every single resource the library owns, including articles
- She knows how databases and articles work but does not see the Find Articles search box in the lower right corner of the page

Design recommendations for this scenario could be --

- Move the Find Articles search box to the left side of the page
- Make the Find Articles header really big and change the color
- Add a Find Databases section to the website

Always distinguish among these three kinds of findings, whether results are conveyed in verbal or in written form.

Verbal reports

The quickest method to communicate findings, and the least formal, is verbal. For this method to succeed, the designers and programmers need to attend the test sessions as either observers or serve as the usability practitioners. Schedule a brief (perhaps thirty minutes) meeting at the conclusion of the last test so that the testing team can immediately summarize the key findings for that round. No written report or video is produced, except the notes individuals chose to scribble to themselves. Everyone shares their observations until the group reaches consensus on the major findings.

Speed is the most positive attribute of this method. On the negative side, there is no written report to share with staff not involved in the process, and over time the findings are subject to memory lapses of the participants. Even with these flaws, this is a great method for experienced staff engaged in very rapid prototyping, perhaps producing a new design iteration each week. Interestingly, it is also a great method for very novice practitioners who need practice drawing significant conclusions from the vast amount of data generated in a test session. During this debriefing session at the conclusion of the tests, team members learn their personal strengths and weaknesses in test interpretation without getting overwhelmed in excessive written reports or learning video editing software.

Written reports

Written reports are popular in libraries. Bulleted lists, charts that record click paths of each user for each test question, and narrative summaries are examples of written report formats suitable for usability tests. Use color-coding to distinguish observation, opinion, and design recommendations. Written reports provide a reference point, useful when a several weeks elapse between design iterations or when anyone questions the purpose of design elements that appear in subsequent design iterations.

On the negative side, reports are time intensive to write and read. Choose a report format that captures the level of detail staff seem to require (it will vary by project and by institution). If staff habitually complain that they lack the time for testing, consider briefer reports. If staff habitually complain that they are not informed of the design work, consider more inclusive reports and save them to a shared file area. In general, discourage long, formal reports that run counter to the spirit of usability testing which is rapid and iterative. Focus on the test experience, not on the report.

Video

Event-recording software is an easy and fairly inexpensive method to create videos of test sessions. Morae, produced by TechSmith, is a popular example of this software.[1] Event-recording software records a video view of the user's facial expression, audio, and desktop activity during the test. This software lets you create one continuous clip of a test session, or string several brief segments together in a storyboard. Files are large; plan a shared file storage system to reduce the need to burn CDs. Remote viewing is a very attractive feature of event-recording software. During remote viewing people can observe tests in a separate room. Observers can easily come and go from the room, eat or drink, and talk amongst themselves without interrupting the test.

Like written reports, these video files are time consuming to create and view. However, a key video clip can convey user behavior better than a thousand written words and is worth the investment once the library clearly intends to incorporate usability into their design process.

Educating staff about usability

While it is obvious that usability findings need to be shared, it is much less obvious that usability practi-

[1] Techsmith. Available: http://www.techsmith.com/ [August 3, 2005].

tioners should be prepared to explain usability theory and methods to staff unfamiliar with usability. These techniques are new in libraries and require explanation. It can take up to a year for an entire library staff to become familiar with the methods, and until that happens plan to repeatedly explain various aspects of it. Staff and stakeholders who understand the underlying methodology are less apt to challenge usability findings. Findings are qualitative and open to interpretation. People unaccustomed to the methodology often misunderstand the work, which can slow down the process and cause unnecessary stress.

The most common criticisms to usability work include distrust that 3-5 users are enough, that paper prototypes are amateurish and childish, and that the detail in a task or test question is faulty, rendering all the findings invalid. All of this can be avoided with a well-designed communication strategy. It is helpful to note that these criticisms are common in any organization, not just libraries.

A multi-layered communication plan works best. Plan some events to provide an overview perspective on usability for all staff, and also plan to address specific issues with smaller audiences. In an overview information session, explain the process. Define specialized vocabulary like *heuristic*, *paper prototypes*, and *iteration*. Provide examples as often as possible. Many usability methods are best explained by example or illustration rather than lecture. Additional proven ways to provide an overview include performing a mock test at a staff meeting. Or, show two consecutive design iterations and discuss the test findings that influenced the second iteration.

One unique area of misunderstanding among librarians regarding usability testing involves the role of the test moderator. Because the test moderator neither teaches nor answers reference questions during a test, reference librarians may question the validity of the role because it is unlike reference librarian behavior. Jeff Rubin, in his book Handbook of Usability Testing,[2] provides a good explanation of the validity of the test moderator role.

Instructional material for usability

Several popular usability websites and books provide instructional material to counter the major common criticisms of usability. This content is particularly useful for smaller audiences. Jakob Nielsen's Alert Box e-newsletter at http://www.useit.com/alertbox/ offers brief explanations of usability basics. Popular newsletter issues include: Usability 101, Return on Investment, and Why You Only Need to Test With 5 Users. OCLC provides a concise treatment of heuristics that includes a chart listing fourteen heuristics and instructions for assigning severity and extent [3]. Steven Krug uses a graphic style to convey many usability tenants in his book Don't Make Me Think [4]. Color photocopies of key pages from his book can persuade unbelievers of the truthfulness that usability works with only 3-5 users. Keep several of these instructional pieces on hand to share at the appropriate moment.

Conclusion

In summary, usability practitioners have several methods to choose from when sharing usability findings. Choose the one that suites your organization and fits with the resources you have allocated for technology development.

Without effective communication of usability findings, usability testing will not impact the website design. A communication plan, including an instructional program for staff, increases the likelyhood of successful web designs.

[2] Rubin, Jeffrey. Handbook of Usability Testing (New York: Wiley) 1994.
[3] OCLC. How we do it: heuristic evaluation. Available: http://www.oclc.org/policies/usability/heuristic/oclc.htm [August 4, 2005]
[4] Krug, Steven. Don't Make Me Think (Indianapolis: Que Publishing) 2000.

Chapter 14. Case Studies

This text includes case studies of usability studies

Purdue University Libraries

Hal P. Kirkwood, Jr., Management and Economics Library, Purdue University

The Purdue Libraries were an early pioneer of Web development, creating a link-rich site prior to the development of Yahoo! Through the late 1990's the site was redesigned only once, and without user input. Anecdotal evidence both by students and library staff illuminated a severe problem with the site. Discussion ensued on the need for a major revision and redesign of the Purdue Libraries Website. Unfortunately there was no specific group in place to take on such an enormous task. The Web Site Support Team was created to oversee the Purdue Libraries website.

The Purdue Libraries team structure was developed in late 1999 - 2000 to facilitate communication across the library system and improve action on the goals and objectives of the Libraries' Strategic Plan. The initial charge to the Web Site Support Team (WSST) was to create an easily navigable, professional, and logically organized web site for the Purdue University Libraries. The Purdue University Libraries consists of 14 departmental libraries with individual webmasters for almost every library. Team members, with a variety of experience and knowledge in web design, were drawn from all over the library system. The team began to plan for a complete revision and redesign of the Libraries' website. A primary consideration from the very beginning was to try and include users; students and faculty, throughout the redesign process.

Preliminary Research

Initially the team developed a base of knowledge among its members by reviewing articles on website redesign and information architecture. The team reviewed writings by Instone, Nielsen, Morville, Rosenfeld, and other respected experts in the field. This brought the entire team up to a base level of knowledge and expertise. The team then proceeded to review, compare, and evaluate similar websites; looking at other comparable university library systems. The team reviewed several other redesign projects as well. A plan was created from this to guide the redesign.

A sub-team of the Web Site Support Team was charged with creating and implementing a heuristic test of the current Libraries' website to compare the existing site to common design elements. The results of the heuristic test showed that the site had not been adhering to many common design elements and constructs.

Online Survey

An online survey was developed to gather basic feedback on user expectations of the Libraries' site and of general user preferences and technical knowledge. Questions were developed by the team and then tested by using student employees within the libraries. The pre-test phase with student employees was extremely beneficial in creating an effective tool. The test phase allowed us to refine the questions to ensure the survey was clear and would provide us with information we could use to help redesign the site. The final survey was then rolled out onto the home page of the Libraries' website. Several of the departmental libraries also included a link to the survey from their home pages. The survey gave us an initial view of our users' concerns with the Libraries' site, a self-perception of their searching abilities, and a scan of their technological position (browser-type, Internet access, etc.)

The online survey, by way of a request for volunteers, also provided the team with a pool of students for the task-oriented testing in the next phase. We also placed several advertisements in the student newspaper requesting volunteers. The incentive was a $10.00 copy card for 20-30 min. of voluntary testing. Our results from the advertisements were very poor; the bulk of our volunteers came from the option to volunteer in the online survey.

Task-oriented Testing

The task-oriented phase consisted of a series of tests whereby students and faculty were observed completing a variety of tasks to help us develop prototypes for the Libraries' site. We again used our student employees to test card sets and questions to ensure they were worded properly and that we would elicit useful information. One round of testing consisted of a card sorting test. The terms and sections of the current Libraries' site were individually placed on cards. The deck of cards was given to a participant and asked to sort the cards into 3 piles; very important, moderately important, and less important. Once this was completed the piles were taken away for later analysis. A second matching deck was given to the participants. They were asked to sort the cards into piles of similar resources that made sense to them. They were also asked to create a label for each pile, using blank cards that were provided. The purpose of these two tests was to determine what was important to the students, what they considered to be related, and what they would call a specific group of resources. This method of testing gave us insight into how the libraries were matching the students' and faculties' expectations of what and how they would find information.

A second round of task-oriented testing consisted of a series of questions that the participants would attempt to answer using the current Libraries' site. A team member would observe them attempting to answer each question and write notes on how they accomplished finding the answer. A completely different group of volunteers was used for this round of testing. The participants were expected to think out loud about their impressions and their considerations while trying to answer the questions. The participants were scheduled in small groups and tested simultaneously in our electronic classroom. The purpose of this test was to determine current navigation problems and to shed light on how students and faculty seek information form the Libraries' site.

Throughout both rounds of task-oriented testing a brief focus group discussion was held after each session. Specifically, the participants attending the card sorting and the task-based testing were brought together in a small group where a general discussion was held about the Libraries' site and Web site navigation in general. An interesting development from this discussion was that the participants often would say they preferred one thing but the testing showed they preferred the exact opposite.

Findings

The results of the task-based testing focused on several areas: terminology, navigation, and site architecture. Terminology problems consisted of difficulty understanding library jargon as well as problems of inconsistent naming of resources and services. Participants commented on having problems with navigating the site, including dead end and orphaned pages. The site architecture was also clearly a problem. Participants reported in the task-based testing and in the card sort testing desiring better, faster access to resources and services. The site had too many layers, forcing multiple clicks to get to useful information.

The team collated all of the information we had gathered and determined that we needed to focus our energy on several specific design elements. The overall look of the site needed to be refreshed and made more appealing. The navigation needed to be more consistent across the entire site including the departmental libraries; this led us to use Cascading Style Sheets to maintain a consistent look with some flexibility for the departmental webmasters. The site architecture needed to be improved drastically with the creation of a controlled vocabulary to be used across the site. Also it was necessary to create clearer paths between pages and sections so visitors know where they are, where they can go, and how to get back. Finally, the original site was far too deep, with an excessive amount of clicking required to find needed resources. Thus the home page needed to become a more prominent portal for linking to services and sources quickly.

Communicating with the webmasters

After collecting all of the data from the task-oriented testing and analyzing it for trends and patterns, the

Web Site Support Team met with the Libraries' webmasters to discuss the findings and to plan for developing the new site. This provided an opportunity for the webmasters to provide input as well as to create buy-in to the redesign process. The Web Site Support Team presented its findings from the testing and engaged the Webmasters in a discussion of the design ramifications of the findings.

Underlying Infrastructure

The next phase was to create the underlying infrastructure, the foundation for the redesign of the Libraries' site. A thesaurus was created to resolve issues of inconsistent terminology and excessive jargon. Site architecture was developed to guide the redesign. Visual elements were discussed and samples were created; leading to the development of prototypes.

Prototypes

The Web Site Support Team then broke into two sub-teams, each assigned to create a prototype of the redesigned Libraries' homepage. It was decided that there was enough variation in the information collected that two separate prototypes should be made and then compared and tested by users. The two teams were each given certain design elements to create the homepage and a sample lower-level page. Once the prototypes were made, the team took them back to the users. Participants in the original card sorting and task-oriented testing were contacted and given access to the prototypes. Minor changes were made based on this feedback. The prototypes were then taken to a new group of task oriented testers. Over a 2-3 week period the team set up laptops in a variety of public locations; several of the departmental libraries on campus as well as several high traffic areas. The team solicited volunteers to sit down and complete several tasks with each prototype and provide commentary on which features they found most useful or implemented most effectively. The team alternated which prototype was tested first to avoid any preference based on testing order. An additional function of this public testing was to promote the Libraries' redesign project and to highlight the fact that we were involving users throughout the process.

Final Redesign

The final redesign consisted of a merged version of the two prototypes. Elements that showed positive results from the final round of testing along with all of the previous testing and research were brought together for the final redesign. A publicity campaign was developed and the final design was presented to the Libraries' faculty and staff. The design was then rolled out publicly to the faculty and students of the university.

At the conclusion we felt very confident that we had designed a more user-friendly and user-centered site. The site received very positive feedback once it was rolled out. We worked in conjunction with Purdue University Marketing so that the color scheme and some design elements were in line with the eventual redesign of the entire University site. Since we actually rolled ours out first it looked like the University was catching up with the Library; this was great for our own marketing and campus standing.

Throughout the redesign process we focused on involving our users: the students and faculty. We knew there were significant problems but we could not just go and make the necessary changes. This had been tried in the past and was questioned by the Libraries administration. Keeping the redesign user-centered and user-involved throughout allowed us to justify every decision we made so that if it were challenged we could go back to the tests and the data to support the direction we had chosen.

Web sites must remain living, adapting entities; thus there is discussion of conducting a new round of usability testing to determine if there are any design issues with the current site. The Web Site Support Team is in discussion with the new Libraries' administration as to the future development of the Website. Issues being debated include possible migration to a content management system and/or outsourcing the next redesign. It is too soon to tell what path the Purdue Libraries will follow, however we will always strive to keep our users involved in the design process.

University of Virginia Library

Leslie Johnston

A culture of assessment

In 2001, the University of Virginia Library was evaluating its methods used for internal assessment and process evaluation. The Library's Web sites provide a key entry point for users to many of the Library's services, thus making it the hub of technology infrastructure. Ease of use and consistency in design and functionality are key ingredients to building a customer-oriented system. With this in mind, a Usability Group with representation from Management Information Systems, Digital Access Services, Communications, and other Library and UVa units was created to undertake assessment of the Library's growing web presence. This group set up the initial criteria for selecting sites to be tested, procedures and forms used in testing, and a metric to quantify testing in an effort to measure success.

Developing usability testing processes

The criteria for sites to be tested were simple. All new sites and any sites scheduled for design revision would be tested. Certain categories of sites -- primarily temporary pages and web-based incarnations of Library exhibits -- would not be tested. Sites in the UVA Library's web sphere are designed centrally by the Communications web staff, but implementation and maintenance are distributed amongst the numerous site owners throughout the Library staff.

It was agreed that, depending upon the site to be tested, either heuristic testing or full usability testing would be selected. Local heuristic principles were laid out (http://www.lib.virginia.edu/usability/heuristics.html), modeled on Jakob Nielson's "Ten Usability Heuristics" (http://www.useit.com/papers/heuristic/heuristic_list.html). When heuristic testing was called for, members of the Usability Group would perform the tests. As for full usability tests, guidelines were developed for creating site test questions, including the creation of site owner questionnaires and templates for test logs http://www.lib.virginia.edu/usability/tests/. Test administration procedures were documented http://staff.lib.virginia.edu/usability/testing_procedures_original.htm.

The full testing process was a relatively simple one. Sites that met the criteria were identified by the web design staff in the Communications unit. In some cases, site owners of sites not due for revision submitted their sites for testing because they saw room for improvement. The owner of each site was asked to fill out a Site Owner questionnaire, to supply contextual information such as the mission, audience, and functions of the site. A set of between ten and fifteen task-based test questions were developed by the Usability Group, or no more than it would take an hour to test. Smaller sites had as few as seven questions. The questions were framed to require the tester to answer a question or perform a task without cluing them in that something definitely is part of the site. The questions avoided the exact language that is used on the site to label the feature or function. The goal was that all questions should be written as "Is there?" or "Can you?" questions, not "Where is x in the site?". Tests are available for review online http://www.lib.virginia.edu/usability/tests/ .

The goal was to test groups of five categories representing undergraduates, graduates, faculty, Library staff, and the public, if appropriate. Depending upon the site, there might be as few as three categories of groups, as small as three students of mixed type, three teaching faculty, and three Library staff. Testers were solicited from the Library staff through email calls for participation. Faculty were solicited through personal requests from librarians. Students were most often employed by the Library, or solicited by student Library employees. Library employees remained on the clock for their tests. Students were paid $10 for their participation.

Card Sorts were also sometimes used as part of the review of an already existing Library site to suggest revisions to a site's structure and labeling. The UVa Library process is a Closed Card Sort. Groups of no more than 6-8 participants were given stacks of cards, one for each of the existing or suggested set of organizing categories and each single site page. The group was asked to organize cards representing pages under the categories. UVa breaks slightly from the usual practice by additionally allowing users to suggest new categories or new wording for category labeling. One or more groups might be tested, representing multiple user constituencies.

Members of the Usability Group administered all tests. Tests were administered with one tester, one facilitator, and one observer to record results on a paper log, logging steps taken by the tester with time elapsed in 15-second intervals. Tester names were never recorded, only the category, such as faculty or graduate student. Test results were summarized and a report written by a member of the Communications web design staff for presentation to the site owner. The recommendations in that report were implemented whenever possible, but time and resources were not always available for site revisions in an environment where site maintenance is distributed. Sites were only on rare occasions re-tested after revisions were made.

The University of Virginia Library developed a set of Balanced Scorecard metrics in 2001 for assessment purposes. A usability metric was put in place with the target that eighty percent of sites that met the criteria should be tested each year. For the first three years the group met or narrowly missed its target.

2005 Review of Usability Procedures

In mid-2005 the Usability group reviewed its procedures. This review was initiated as part of an overall review of procedures for all activities related to internal processes with corresponding Balanced Scorecard metrics. The metric had previously been grouped with a category of measurable activities called "Learning & Growth," looking at how well the library was positioned to ensure that goals are met in the future. Usability testing and its metric are now reviewed in the context of how well the library's internal processes function to efficiently deliver library collections and services.

Recommendation: Most processes aren't broken, so don't fix them

The criteria for identifying sites remain the same. Heuristic principles remain unchanged. Test development guidelines and selection of categories of users to be tested will not change.

Recommendation: Change test recording procedures

Review of the observation and logging of test results by multiple individuals showed that methods for recording tester activities, the level of detail recorded, and the consistency of time logging varied widely. To create a more consistent recording environment, the Library is switching from manual recording to the use of digital cameras and Morae software to record testing activities.

Recommendation: Change the definition of success

What changed most drastically is the metric for success. Simply ensuring that sites were tested did not assure that testing brought about improvement in usability. Success is redefined as a documented improvement in the usability of a site.

Recommendation: Some procedures must be changed to better measure success

To better measure improvement, changes were needed in both the Library's testing process and site development procedures. The old process was one where a site was tested, the test reported upon to the site owner, and changes were often made (but not always), and updated sites were rarely re-tested. The new process is one where a site is tested, the test results are reported to the site owner, recommendations are reviewed with the web design staff in the Communications department, implementation of needed changes is required, and the site will always be re-tested after it is revised.

The UVA Library has not changed the guidelines for developing usability test tasks and the framing of questions, to continue to get the needed type of input from the tests. The Library has instead identified an improved process for quantifying a measurement of success. The revised metric is a comparison between the first and the second test, where the percentage of right answers found by the subject on the first try (regardless of the path they followed) is compared. Results are quantified by counting the number of right answers and success measured through an improved percentage of right answers in the retesting.

While essential testing procedures have not changed, documentation of the procedures has been revised and expanded to now include guidelines for all types of tests -- heuristics, card sorts, and full tests -- as well as for the development of tests and test follow-up http://staff.lib.virginia.edu/usability/testing_procedures.htm. The earlier version focused solely on test administration.

Recommendation: Usability has to be better institutionalized as part of the Library's operations

This newly revised procedure also requires administrative and staff buy-in for changes in the Library's site development process. Usability testing is now an integral part of the site development process, and Library site owners must now take seriously the recommendations made based on usability testing. New documentation is under development to codify standards for setting up a UVa Library site, and identifying the process for building a site that includes the involvement of (and accountability to) the Communications department and the Usability Group.

MyLibrary and Campus Portal: a case study

Vishwam Annam
Alison Aldrich, Wright State University Libraries

Abstract

Purpose - To provide an overview of the process involved in usability testing with a case study about offering library services through a campus portal.

Design/methodology/approach - To gather feedback from potential users of a customizable electronic library product, participants were selected from different library patron groups: undergraduate, graduate students, faculty, and library reference staff. The tests were conducted as one-on-one structured interviews. Participants were asked to perform selected tasks by using our campus portal (we call it WINGS) and another University's portal. Responses were noted and tallied manually rather than through use of computer software because of the relatively small number of people involved.

Findings - Responses about the potential usefulness of a customizable library were mixed. Participants indicated that having a customizable library in the campus portal may be useful. A majority of the participants were using WINGS for one or two functions like webmail, calendar, Course Studio, etc. but were not utilizing all of the portal's features.

Keywords: Library portals, library services, case studies

Introduction and Background:

Wright State University implemented SunGard SCT's Luminis as our campus portal in 2004. The portal, which we call WINGS, provides single sign-on access to University services like course registration, grades, accounts, campus activities, and more. We see tremendous potential for WINGS to make the Wright State University community's work easier.

The University Libraries created several optional channels for WINGS, including channels for the catalog, course reserves, dictionary searching, and library news. These channels link to static pages on our library website. The channels do not automatically display personalized information upon portal login. We would like for users to be able to customize lists of resources (such as databases and journals) that they use most frequently, and ultimately we would like library services such as requesting books, checking account status, and interlibrary loan to work under the single sign on protocol for WINGS. Since the portal is designed to manage multiple accounts, we are investigating different ways to create a customizable electronic library within WINGS.

Before implementing a customizable library, we conducted a small usability study in order to gauge our patrons' response to the product. Our goals were to gather opinions from users about the library channels we currently offer through WINGS, and the potential usefulness of a customizable electronic library product.

In this essay, we will describe the procedure we used, the conclusions we reached, and how the results will inform our future decisions about customizable library implementation.

Selecting Participants

According to Jakob Nielsen's popular rule of five, 85% of a site's problems can be found with as few as five participants. It may be true that you can learn a lot from just a few interviews, but when the audience for your product is composed of distinct groups of people, it is good idea to select at least two participants from each group. In our study, the groups were undergraduate students, graduate students, faculty and staff of Wright State University. We selected two undergraduate students and two graduate students. Due to availability constraints, we were only able to test one faculty member and one library reference staff member. We chose participants based on our personal contacts, attempting to find representatives from multiple fields of study. Tests were scheduled by e-mail or by phone. We made note of the demographics (status, field of study) for each confirmed participant.

The Script

We used the same set of open-ended questions for all tests in order to clearly capture the different perspectives of our different audience groups. Our script consisted of three sections. In the first section, we asked participants to characterize their use of our library website:

1. Section 1:

   Have you ever used the Libraries' website before? If so, for what purpose?

2. How frequently do you use the Libraries' website?

3. Do you bookmark the links you frequently use, or do you go through Libraries' site every time?

4. If you had just the links you use collected in one place, instead of going through Libraries' site every time to find them, would that help you?

   Section 2:

   In the second section, we asked participants to describe their use of WINGS and the library channels currently available within WINGS:

5. Have you ever used WINGS?

6. When was the first time you used WINGS? When was the last time you used it, and for what purpose?

7. How frequently you use WINGS? What is your primary reason for using it?

8. What do you think is the main purpose of WINGS?

9. Did you know that the Libraries offer some services through WINGS?

   If yes, how did you come to know about these services

   If no, what services would you expect the Libraries to offer through WINGS?

10. Where in WINGS would you expect to find these services.

11. Once they locate the Libraries' channels on the Academics tab of WINGS, give them some time to look at the page. Are you able to find most of the features that you normally use on the Libraries' site?

    Section 3:

    In this last section, we used a guest login to demonstrate MyLibrary as implemented in Lehigh University's campus portal. We asked participants to imagine how they might use customizable library services as a feature of WINGS:

12. We are thinking about doing something like this ... show them MyLibrary as implemented through Lehigh's portal. Give them ample time to look at the site.

13. What is your first impression on MyLibrary?

14. You can customize your electronic resources by following some simple steps. Do you want to try it? You can bookmark your favorite links, and do many things as you would with the Libraries' website.

15. What MyLibrary features you would like to see in our WINGS?

16. What group of people do you think would find MyLibrary most useful?

17. Would you use WINGS more often if something like MyLibrary were in place?

18. Do you have any other ideas or recommendations?

Conducting the Usability Tests

The test monitor plays a crucial role in the success of any usability test. His or her role is to lead the participant through the test questions while being careful to not skew the results in any way. We begin each usability test by thanking the participant for volunteering, then reassuring the participant that the test is of our site, not of them.

The usability tests described here were conducted in a much less formal fashion than usability tests we have conducted on our library website. The current project (implementation of a customizable library) is still in the initial planning phases. Comparatively few participants were involved, and we just wanted to get a general feel for what they had to say.

The testing locations were different for each participant and were selected by the participants themselves for comfort, convenience, and so as to take a minimum of their time. The tests could be conducted from any computer with Internet access. With a more formal test, it is more important to standardize the experience, conducting all tests from the same location. When we conducted tests of our library website, we tested from computers that had Camtasia installed for recording mouse clicks and vocal comments.

Results summary

Our subjects were two undergraduate students, two graduate students, one faculty member, and one library reference staff member. Participants' levels of experience with the Libraries' website ranged from basic to advanced. The undergraduates we surveyed had used the site for course reserves and catalog searching but had little experience with article databases. Graduate students and faculty were more sophisticated in their use of the website. They generally reported bookmarking frequently used pages such as the electronic journals list. The librarian did not bookmark pages because he was familiar enough with the site to navigate through it quickly as needed.

The second part of our survey addressed participants' use of the WINGS portal. Of the six people we surveyed, only one could be considered a regular WINGS user. This person, a graduate student, reported using WINGS about once a week, mainly for email, the calendar, and the academics tab. She had done some customization of her WINGS portal and was the only participant who knew about the library channels we created under the Academics tab. She supported leaving the tab as it was, with the library integrated with Academics. One undergraduate student had used the Course Studio function within WINGS. Most participants used WINGS very occasionally for e-mail, preferring an alternative URL for accessing Wright State e-mail via the Web. The library reference staff member was required to use WINGS to maintain his professional calendar and to access group messages and files, but was dissatisfied with the efficiency of those processes.

The third part of our survey involved a demonstration of the MyLibrary system as implemented at another university. Reaction to MyLibrary was mixed. One graduate student was very enthusiastic about the customization options and said that MyLibrary would increase her use of WINGS. Most other participants acknowledged that it was nice that customization options existed, but could not commit to saying MyLibrary was something they would use personally. The mechanics of customization seemed straightforward to the participants. However, the faculty member expressed a lack of confidence in his ability to choose the best customization options for himself. He worried about "missing something" and was pleased to hear that resources could be recommended for him by the subject liaison for his department. He also expressed reluctance to change his research habits unless MyLibrary made his work significantly easier.

The library reference staff member did not think students would be likely to use MyLibrary. The faculty member, on the other hand, commented that MyLibrary would probably get most of its use from people without ingrained research habits.

Major Findings and Recommendations

- Participants have not used the WINGS portal regularly.

  WINGS is still fairly new, having first been released in 2004. We realize that, as WINGS offers more services and a wider variety of content in future, more people are going to use it regularly.

- Some participants could not find library services in WINGS.

  Since library links in the portal are currently under the Academics tab, they usually go unnoticed. The links are only visible to users when they click on this tab. Since we are planning to expand the library services available through WINGS, the library could have its own tab which would be visible from most pages within WINGS.

- Most participants prefer having links to library resources by default rather than customizing their own lists.

  Resources can be recommended for each patron group (undergraduate, graduate, staff, and faculty) by the subject liaisons for each academic department. Patrons, based on their demographics, will see links that are already somewhat customized the first time they log in to the customizable library in WINGS.

Conclusion

This case study illustrates that usability testing can and should be done early on in the web development process. By conducting just a few informal interviews, we gained a better understanding of how our patrons use the WINGS portal and what customized library services they would like to see in it.

Students, faculty, and staff have different library use patterns. As such, we are thinking about developing a customizable library system in which the default display varies based on patron type. Since this system would be homegrown, it would be easier for us to maintain and troubleshoot. The WINGS portal is still in its introduction phase on our campus, but we anticipate that more people will use it regularly in the future. We are continuing with our plans to implement the customizable library within WINGS. As the project continues, we will conduct additional, more formal usability tests.

Bibliography

References:

Authentic Behavior in User Testing http://www.useit.com/alertbox/20050214.html

Recruiting Test Participants for Usability Studies http://www.useit.com/alertbox/20030120.html

Genuis, Shelagh. Web Site Usability Testing: A Critical Tool for Libraries. Feliciter, v.50, no.4 (2004), p. 161-164.

Hartson, H. Rex et al. Usability inspection of digital libraries: a case study. International Journal of Digital Libraries, v.4, no.2 (2004), p. 108-123

Library participation in campus web portals: an initial survey Reference Services Review, Year: 2005 Volume: 33 Number: 2 Page: 144 -- 160

Developer Center Article http://www.macromedia.com/devnet/articles/usability_testing.html

Guide to Planning and Conducting Usability Tests http://www.lib.washington.edu/usability/guidelines/usability-guidelines.html

Usability testing http://www.cu.edu/irm/stds/usability/

Websites we tested:

Wright Info, News and Group Services(WINGS) http://wings.wright.edu/

Wright State University Libraries http://www.libraries.wright.edu/

Lehigh University portal http://portal.lehigh.edu

Part VI. Underlying technologies

Table of Contents

Chapter 15. What is XML, and Why Should I Care?

Tod A. Olson, The University of Chicago <tod@uchicago.edu>

Since its introduction in 1998, XML has become a well-established technology, having gained widespread adoption for the storing and exchanging of information, including in the library community. This essay will introduce the reader to XML, explain its syntax and how it is used as a means of encoding information, and follow up with why it is important to the library community.

What is XML?

The eXtensible Markup Language (XML) provides a general-purpose syntax for encoding information. XML is designed to be easily processed by machine, and yet human-readable, while addressing practical concerns of building Web-based applications, and drawing on experiences with the earlier Standard Generalized Markup Language (SGML). XML is a Recommendation of the World Wide Web Consortium (W3C), the 1.0 recommendation is available at http://www.w3.org/TR/2004/REC-xml-20040204/. The W3C develops the technologies used by the Web. More information about W3C and its activities is at http://www.w3.org/.

A descriptive markup language

XML is a descriptive markup language. It provides a syntax that lets us add descriptive tags to an electronic text document. Take a number like 1966. On it's own, it could represent a number of things, possibly a year, street address, or a mortgage payment. When we add descriptive tags to the number, its role becomes clear:

```
<date>1966</date>
```

We can treat `<date>`1966`</date>` appropriately in our processing, whether indexing, exchanging data with another system or formatting for display to a person. In XML terms, `date` is an example of an *element*.

While XML defines a syntax for marking up or tagging information in documents, it does not define a list of allowed tags. Instead, a document author is free to invent the tags that are needed. There are, however, mechanisms for declaring exactly what tags may appear in a document. Declaring what tags may appear defines a markup language. For this reason, XML is a metalanguage; it is a language for defining markup languages.

For more on text markup, see Coombs, Renear, and DeRose (1987).

Structure in XML: elements and attributes

Elements provide the structure of information in XML; they contain and describe information. An element is marked by a start tag and an end tag. The start and end tags look like `<elt>` and `</elt>`, where `elt` is the element name. Any text between them is the element content. In the above example, "date" is an element, and the tags `<date>` and `</date>` mark the text "1966" as a "date".

Elements can contain text characters and other elements. For example:

```
<para>An <card>Ace</card> beats a <card>King</card>.</para>
```

or

```
<originInfo>
  <place>New York</place>
  <publisher>Bantam</publisher>
  <dateIssued>1971</dateIssued>
  <copyrightDate>1968</copyrightDate>
</originInfo>
```

The nesting of elements allows a hierarchical document structure. Any information that can be modeled as a hierarchy can be represented in XML.

Element names are case-sensitive. For example, `<place>` and `<Place>` are start tags for two different elements, and must be closed by `</place>` and `</Place>` respectively.

Elements may have *attributes*, which may be thought of as metadata for the element. An attribute appears in the opening tag of an element as a name, an equals sign, and a value in quotes. Whitespace around the equals sign is allowed. Either single or double quotes may be used. A value begins with the quote character and continues until the next occurrence of the same quote character. For example, in the Metadata Object Description Schema (MODS) `title` element, the `type` attribute can be used to specify what kind of title is being tagged. Here the uniform title is distinguished from the "default" title on the title page of a piano score:

```
<titleInfo type="uniform">
    <title>Waltzes, piano</title>
    <partNumber>op. 34</partNumber>
</titleInfo>
<titleInfo>
    <title>Trois valses brillantes</title>
    <subTitle>pour le piano : op. 34, no. 1-3</subTitle>
</titleInfo>
```

Similarly, in MODS a personal name can be distinguished from a corporate name, and the parts of a name can also be distinguished:

```
<name type="personal">
  <namePart type="family">Chopin</namePart>
  <namePart type="given">Frédéric</namePart>
  <namePart type="date">1810-1849</namePart>
</name>
```

Elements may have more than one attribute, but the attributes must all have different names.

XML requires that there be only one *root element* in a document, one top-level element that contains all of the document content, including all other elements. That is, elements nest in a hierarchy, and XML wants a document to have only one hierarchy. Looking at the two sample XML fragments immediately above, the first fragment could not stand alone as a legal XML document, because there are two titleInfo elements at the top level, with no element containing them both. The second fragment is a legal XML document, because the name element acts as a single root element.

Some elements have no content, but are meaningful just by existing at a certain place in a document. If an element has no content, it may use the empty tag syntax: <*elt* />. For example, in Text Encoding Initiative (TEI), used for scholarly markup of literary texts, often of older printed materials, <lb/> and <pb/> shows where a line break or page break was in an original print edition of a document, just as <br/> shows a line break in XHTML. (Space after the element name is permitted, but not required; <br/> and <br /> are equivalent.)

Empty elements are often used with attributes. In TEI, the ptr can be used to make simple cross-references within a document. ptr is always empty, but it has a target attribute which contains a unique identifier for the target of the cross-reference. In XHTML, the link element is empty, but uses attributes to describe and locate the document that is linked to. In the following example, the link attributes tell a Web browser where to find some external document (href), that this document is a stylesheet for the current XHTML document (rel), and that the stylesheet is written in CSS (type):

```
<link href='mystylesheet.css' rel='stylesheet' type='text/css' />
```

Closing tags required: making structure explicit

If the XML syntax seems similar to HTML, it is because HTML and XML share some common ancestry: HTML is an application of the Standard General Markup Language (SGML), and XML began as a simplified version of SGML. Unlike HTML (and SGML), XML requires closing tags, and empty tags have special syntax. This makes it easy to process XML documents, whether as files or data streams, without special knowledge of the particular vocabulary employed. Consider this example from HTML:

```
<ul>
<li>A list item
<ul>
<li>Another list item
```

Is the second list an entirely separate list from the first, or a sub-list? Without closing tags, the structure is implicit, and requires special knowledge of HTML to process. There is a similar issue with empty tags in HTML: a <br> tag is empty, it does not begin a new element, and there will never be a </br>. This is a special property of the br element in HTML. There is no syntactic clue to help us; we need special knowledge of br in order to understand the structure of any document that contains it. So we see that the document structure is not determined entirely through syntax, but depends on the semantics of the elements.

By requiring closing tags, XML forces the document author to be explicit. In XHTML, the author will write

```
<ul>
  <li>A list item</li>
</ul>
<ul>
  <li>An item in a separate list</li>
</ul>
```

or

```
<ul>
  <li>A list item
    <ul>
```

```
      <li>An item in a sub-list</li>
    </ul>
  </li>
</ul>
```

The document structure is unambiguous, even if you don't know the first thing about XHTML. Similarly, if `<br/>` appears in our XHTML document, we know exactly what its effect on the document structure is: it is an element with no content. No special knowledge of the `br` element is required in XHTML. The document structure in XML is determined unambigously by the syntax, and is not dependent on the element semantics.

Entity References

An *entity* can be thought of as a piece of text that has a name. An *entity reference* allows us to refer to that text by name and use it in a document.

We have seen that some characters have special meaning in the XML syntax. For example "<" always signals the beginning of an element tag. The character "<" is illegal in element content because it would complicate parsing, as it would be impossible for the software to determine whether the "<" starts an element or is just a character. For those cases where we need to use "<" as part of our data, XML provides a *predefined entity reference*, "&lt;", where the name "lt" is mnemonic for less than. An entity reference begins with an ampersand, ends with a colon, with the entity name in between. The ampersand therefore is also a special character, and we must always use "&amp;" when an ampersand is part of the element content or an attribute value. There are only five such predefined entities, and they are defined for the characters that signal elements, attributes, and entities, and they appear below:

Character	Name	Entity reference
<	less than	`&lt;`
>	greater than	`&gt;`
"	quote, or double quote	`&quot;`
'	apostrophe, or single quote	`&apos;`
&	ampersand	`&amp;`

There are also *character entities* for referring to any character by its decimal or hexadecimal value. For the section sign, §, the character entity references would be `&#167;` and `&#xA7;`, respectively. This is useful when input mechanisms do not allow a character to be entered directly, or to be certain that the document can be deciphered by humans working in computing environments which do not handle Unicode well.

Document Type Definitions (DTDs), discussed briefly below, allow the definition of custom entities above and beyond the predefined entities and character entities. This can be used to name and refer to text that may be referred to repeatedly in the text. This may be done to ensure that some string, such as a specific name or phrase, is always repeated exactly, or for some other reason. The DocBook DTD, for example, defines a large number of entities that give mnemonic names to characters that are not always easy to enter from the keyboard, such as the `&copy;`, which is replaced during processing with the copyright symbol, ©.

XML declarations

An *XML declaration* gives information about the XML version, and possibly the character encoding, used by the document. The following XML declaration asserts that the document conforms to XML 1.0, and that characters are encoded in UTF-8, the 8-bit Unicode encoding:

```
<?xml version="1.0" encoding="UTF-8"?>
```

(By default, XML documents use the Unicode character set, in particular, the UTF-8 encoding. This allows XML documents to carry information in many languages.)

The XML declaration allows us to be certain that a document really is XML, and not SGML or some other similar-looking format. According to the XML 1.0 Recommendation, § 2.8, an XML document should have an XML declaration, and if an XML declaration is present, it must be the first statement in the document.

Processing instructions

Processing instructions allow instructions to specific XML systems to be embedded in XML documents. Processing instructions follow this pattern:

```
<?target instruction?>
```

where *`target`* is a name that a specific processing system will recognize, and *`instruction`* is a sequence of text that tells the processing system to take some action. Processing systems that do not recognize a target name will ignore that instruction. For example, the following XML fragment is meant to be processed by a PHP system:

```
<p>Today is <?php echo date("M d, Y")?>.</p>
```

PHP will recognize the target of the processing instruction, `php`, and follow the instructions: it will get the current date, format it, and place the result into the document where the processing instruction was. The output will be something like this:

```
<p>Today is Jan 23, 2006.</p>
```

Taget names that begin with “`XML`”, in any combination of upper or lower case, are reserved. These target names may be used by the XML specification or for standardization. For example, the XML specification defines the target `xml` to be the XMl declaration. As another example, `xml-stylesheet` is a standardized processing instruction which allows a stylesheet to be associated with an XML document. Here, a CSS stylesheet is associated with the document:

```
<?xml-stylesheet type="text/css" href="mystyle.css"?>
```

For details on `xml-stylesheet`, see http://www.w3.org/TR/xml-stylesheet/.

Comments

Comments in XML documents mark some part of the text as not being part of the document's data or structure, and which are to be ignored during processing.[1] Comments are useful for a variety of reasons, such as recording commentary for people who may edit the document in the future, or for temporarily “turning off” a part of the document, perhaps for testing purposes. Comments begin with `<!--` and end with `-->`, as follows:

```
<!-- This is a comment ... -->

<!-- <para>... and this paragraph has been commented out.</para> -->
```

The string `--` is not allowed within a comment. SGML uses this notation to structure comments in a certain way; XML rejects this extra complexity for comments. XML forbids this notation within comments to preserve compatability with SGML processors, where it can cause errors if the precise structure is not observed.

CDATA sections

Character data is any data in an XML document that is not part of the markup. Mostly, this refers to the text content of elements, but not the element tags themselves, entity references, comments, processing instructions, or the like. A *CDATA section* marks the enclosed block of text as character data in which there is no markup and all characters are treated literally, that is, there are no elements and no entites. Put another way, inside a CDATA section, the special meanings of “<” and “&” are turned off. A CDATA section begins with `<![CDATA[` and ends with `]]>`.

CDATA sections can be very useful when some of the text looks like markup but should not be interpreted as markup, or uses notations with “<” or “&”. In the XML source for this essay, many of the examples of XML markup are enclosed in CDATA sections. For example, the example in the section called “Processing instructions” is coded as follows:

```
<![CDATA[<p>Today is <?php echo date("M d, Y")?>.</p>]]>
```

Well-formed documents

A document is *well-formed* if the basic syntax and nesting rules are followed. The most important of these rules are:

1. An XML document has a single root element.
2. Elements must be closed; elements must be marked with a start tag and a matching end tag, or by an empty tag.
3. Elements nest properly. A child element must be entirely contained within its parent, partial overlap with other elements is not allowed.
4. An element may not have multiple attributes of the same name.
5. Attribute values must be quoted.
6. “<” and “&” must not appear directly as characters in element content or attribute values, but must be represented by their entity references.

[1] Some XML processors make it possible to examine the contents of comments, but they are not required to.

The rules for well-formedness promote clarity and unambiguous encoding of information in XML documents, and make it relatively easy for software to parse an XML document.

To explore this notion briefly, consider the fragment below, which violates the rule 6 above by having an unencoded < in an element's character data:

```
<rule>a<b and b<c implies a<c</rule>
```

Any XML software will expect "<" to signal the beginning of a new element. "<b and" will look like part of an opening tag with an attribute, but the equals sign required in an attribute never appears. If "<" were allowed here, any software parsing this XML would have to decide from the characters that follow whether the "<" is the beginning of a tag or just more text. This would add unnecessary complexity to the software, violating the principle that XML parsers be straightforward to implement. This rule is similar to the requirement that tags be closed, which we examined earlier, in that it helps to ensure that the document structure is unambiguous and easy to for software to determine.

XML vocabularies: schemas, namespaces, and validation

While XML itself does not define a set of elements, it can be useful to formally define the vocabulary of allowed elements in a document, including each element's attributes and its content model (what the element may contain: other elements and/or text). Formal definition of the vocabulary helps to establish a shared understanding of the documents. Often a community will have a set of common needs, and having an agreed upon, formally defined vocabulary helps meet these needs. This is especially true if exchanging the documents is important.[2] If a vocabulary is formalized (the elements, attributes, and content models expressed) in a machine-readable way, the document can be validated, that is, it can be checked to ensure that it conforms to the formalized vocabulary.

Sometimes, we will also want to combine elements from different vocabularies in the same document. Usually, a vocabulary is designed for a specific problem domain, but sometimes the information we need to record cuts across more than one domain. For example, METS is a vocabulary that defines structural metadata for electronic objects in digital libraries, but relies on other vocabularies for descriptive metadata.

To motivate the need for formal XML vocabularies, consider the following fictitious invoice:

```
<?xml version="1.0" ?>
<invoice>
    <vendorId>1324123</vendorId>
    <account>6393487</account>
    <item>
        <descr>A Spectre is Haunting Texas, by Fritz Leiber (pbk)</descr>
        <price currency="us">6.95</price>
    </item>
</invoice>
```

Some software would probably act on this document to verify vendor IDs, check account numbers, deduct money from an account and send it to the vendor. So the producer of the document and the recipient must have a shared understanding of the elements and the document structure. If the producer makes a change to the elements or structure of the document, this has consequences for the recipient. Valida-

[2] The MARC format for bibliographic records is an excellent example of this idea. MARC development was motivated by the desire to share certain documents, biblographic records, among libraries; its success made large-scale cooperative cataloging possible.

tion helps to ensure that all parties adhere to the shared understanding of the document.

The mechanisms for formalizing XML vocabularies are generally called schemas. There are three major XML schema languages that are important in the library world: Document Type Definition (DTD), XML Schema and RELAX NG. DTDs are an older way of declaring elements and their attributes content models, inherited from SGML. They are flexible, modular, and well understood, but use their own special syntax rather than XML syntax. XML Schema and RELAX NG use XML syntax for these declarations. All three are well supported by XML editors and processing tools, and are used to develop important XML vocabularies.

XML documents may specify which vocabulary they adhere to. A document adhering to the DocBook DTD, for example, may look like this:

```
<?xml version="1.0" ?>
<!DOCTYPE chapter PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
                         "http://www.docbook.org/xml/4.4/docbookx.dtd"[
<!ENTITY % guimenu.module "IGNORE">
]>
<chapter>
   <title>What is XML and Why Should I Care?</title>
   <abstract>
      <para>Since its introduction in 1998, XML has become...</para>
   </abstract>
   <section>
     <title>What is XML?</title>
     <para>The eXtensible Markup Language (XML) provides a general-purpose syntax f
   </section>
</chapter>
```

This document begins with the XML declaration. Next is the *document type declaration* which specifies `chapter` as the root element, gives the public identifying string for the DocBook version 4.4 DTD, and shows where to find the DTD so that the document may be checked for validity. DTDs can be written to be very modular, with components that can be turned on or off, new components added, or existing components modified. *Parameter entities* are used to customize DTDs. In this example, the parameter entity `guimenu.module` is used to disable the DocBook module that defines markup for writing about GUI menus. There are many such DocBook modules for marking up specific textual features that may be enabled, disabled, or modified using parameter entities. New elements and attributes may be defined and integrated into the content model. This should hint at the power and flexibility of expression that can be retained even when the vocabulary has been formalized. DTDs use a syntax that does not resemble the rest of XML, but is retained from SGML.

XML Schema and RELAX NG have different modularity and extensibility mechanisms, which will not be discussed here. They do not require a `DOCTYPE` declaration, but use a different mechanism, namespaces, for identifying their schema, as shown below. They do not allow the definition of entities, but do allow more sophisticated control over element content, such as specifiying that a element may only contain numeric data.

Namespaces use a Uniform Resource Identifier (URI) to identify the vocabularies to which elements and attributes in a document belong. A *namespace declaration* is used to associate the namespace URI with a prefix, and that prefix can be applied to elements and attributes, marking them as belonging to the namespace. The syntax for the namespace declaration is "`xmlns:`", the prefix, "`=`", and the namespace URI in quotes, as in this template:

```
xmlns:prefix="URI"
```

A namespace declaration appears as an attribute to an element. The prefix is defined for use with the current element and any of its attributes or content. If the prefix is omitted from the namespace declaration

```
xmlns="URI"
```

the URI identifies the default namespace for the current element and all elements and attributes it contains. Any element or attribute that is not prefixed is assumed to be part of the namespace identified by the URI. The URI for any vocabulary is typically defined as part of the formalization of that vocabulary.

For example, both examples below are equivalent in that the mods element is identified as being part of the MODS v.3 namespace:

```
<mods:mods xmlns:mods="http://www.loc.gov/mods/v3">…</mods:mods>
```

```
<mods xmlns ="http://www.loc.gov/mods/v3">…</mods>
```

If two elements in different vocabularies have the same name, no problem, they will have prefixes associated with different URIs. For example, consider the following document:

```
<?xml version = "1.0" encoding = "UTF-8"?>
<mets:mets xmlns:mets="http://www.loc.gov/METS/"
  xmlns:mods="http://www.loc.gov/mods/v3"
  xmlns:dc="http://purl.org/dc/elements/1.1/">
  <mets:dmdSec ID="HZN5117992MODS">
    <mets:mdWrap MDTYPE="MODS">
      <mets:xmlData>
        <mods:mods>
          <mods:titleInfo>
            <mods:title>Marche funèbre</mods:title>
            <mods:subTitle>tiré de la sonate</mods:subTitle>
          </mods:titleInfo>
          <mods:name type="personal">
            <mods:namePart>Chopin, Frédéric</mods:namePart>
            <mods:namePart type="date">1810-1849</mods:namePart>
          </mods:name>
        </mods:mods>
      </mets:xmlData>
    </mets:mdWrap>
  </mets:dmdSec>
  <mets:dmdSec ID="HZN5117992DC">
    <mets:mdWrap MDTYPE="DC">
      <mets:xmlData>
        <dc:title>Marche funèbre : tiré de la sonate</dc:title>
        <dc:creator>Chopin, Frédéric, 1810-1849</dc:creator>
      </mets:xmlData>
    </mets:mdWrap>
  </mets:dmdSec>
  <mets:structMap TYPE="physical">
    <mets:div TYPE="score" DMDID="HZN5117992MODS HZN5117992DC">
      <mets:div ORDER="1" ORDERLABEL="" LABEL="Cover with title" TYPE="page">...</
      <mets:div ORDER="2" ORDERLABEL="2" LABEL="" TYPE="page">...</mets:div>
    </mets:div>
  </mets:structMap>
```

```
</mets:mets>
```

This METS document is adapted from one used to represent a piano score as a digital object. The two `mets:dmdSec` elements contain descriptive metadata as defined by schemas external to METS, specifically MODS and Dublin Core. The `mets:structMap` element contains structural metadata for the digital score object, which is METS' specialty. The file section, with images of the score, is omitted for brevity.

Consider the namespace declarations in the `mets:mets` element:

```
xmlns:mets="http://www.loc.gov/METS/"
xmlns:mods="http://www.loc.gov/mods/v3"
xmlns:dc="http://purl.org/dc/elements/1.1/"
```

The prefix mets is bound to the URI http://www.loc.gov/METS/, the METS namespace. Simlarly, the prefix mods is bound to the URI for the MODS namespace, and dc to the URI for the Dublin Core namespace. By using these namespace prefixes, each element is unambiguously identified as belonging to a specific namespace. Grouping elements by namespace helps software know how to process the elements. During validation, for example, the software can recognize all elements that will be found in a schema describing namespace identified by http://www.loc.gov/METS/, the METS namespace. Namespaces also distinguish between elements of the same name defined in different vocabularies. MODS and Dublin Core both define elements named "`title`", but which have different meanings. By declaring namespaces for MODS and Dublin Core, and using the namespace prefixes, `mods:title` and `dc:title` are clearly distinct.

We can even specify where to find a schema document. The XML Schema vocabulary, for example, defines an attribute, schemaLocation, that associates a namespace URI with a location where the schema definition can be found. For example, adding the following attributes to `mets:mets` would declare the XML Schema namespace and use that namespace to express where to find the MODS v.3 schema:

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.loc.gov/mods/v3 http://www.loc.gov/standards/mods/v
```

Be aware that DTDs predate namespaces, and are not namespace aware. If namespaces are important for your information, you will be dealing with XML Schema or RELAX NG, not with DTDs.

Advantages of XML

Naturally, proponents of XML like to point out the advantages of XML as a data format. This essay will touch only on a few of these advantages; more thorough discussions may easily be found in the materials listed at the end of this essay.

XML, as an information container, is system independent. XML was designed to be processed by a wide variety of systems, and is often used to communicate information between systems. In part this system independence comes from XML's being a text format, not binary format, with a very simple, dependable syntax and structure. A simple text editor is all you need to write an XML document, though a modern Web browser will help check that XML documents are well-formed, and possibly that they are valid. Contrast that to the binary format of popular word processors, where the ability to read and write a document relies on software with an awareness of the binary format.

XML is extensible. Because its creators knew that they could not determine all possible uses of XML, XML does not define a specific tagset. Instead, it allows one to define the elements needed for a particular situation. Even formalized vocabularies can be designed to be customized.

Finally, XML separates semantics from presentation. This advantage of XML comes from its being a descriptive markup language. Consider the first example:

```
<date>1966</date>
```

By marking the role of a part of the document, we can index it to support searching by dates, or we can format it in some special way when printing or displaying it on a screen. This may seem typical when the XML document resembles a set of data fields, such as a the invoice example. But XML also accommodates prose-like journal articles, where the benefits of driving both searching and presentation from the same markup tag are more apparent. The searching granularity is limited only by what we are willing to mark up in the documents; if desired, each document may be treated as a little database. And because presentation can be driven by the role of any component, we have tremendous flexibility in presentation. The date above, or title, etc., could be displayed in bold, in a special font, or just the same as any surrounding text, just by recognizing the `date` tags, and all of the documents in a collection could use the same presentation. The presentation can be tailored to the specific use, for example, using different fonts in print and online versions, or the presentation can be changed to match the latest look of your institution's website. All of this flexibility is available with no need to edit the underlying XML.

Companion technologies

XML has a number of companion technologies for manipulating and otherwise operating on XML documents.

There are standard application programming interfaces (APIs) for XML. Applications need to extract information from XML documents for use in a variety of ways, often for display to the user or in commmunication with other applications. The XML APIs allow the programmer a standard way to extract this information. An application can load a module that implements the API, and the programmer can rely on that module to parse the XML documents without the programmer having to implement the XML parser directly.

One of the earliest XML APIs is the Simple API for XML (SAX). SAX uses an event-based model, viewing XML documents as a linear sequence of opening and closing tags, text, and the like. A SAX-based application registers special code to be invoked when the different types of XML events are encountered. SAX is fast and does not require the whole document to be read into memory, but it is up to the application to keep track of the document structure. SAX can be difficult to use for complex processing, especially if multiple passes over the document is required. SAX was developed for Java, but has been used as a model for XML parsing in other languages.

The Document Object Model (DOM) is a tree-based programming interface for XML defined by the W3C. The DOM stores the entire the document in memory represented as a hierarchy of nodes. There is a root document node, and under this is a hierarchy of nodes representing every element, attribute, text segment and the like. The DOM provides mechanisms to navigate this hierarchy. The DOM also allows for document node structures to be created or modified in memory and then output as XML. By representing the entire document structure in memory and providing mechanisms to navigate that structure, the DOM makes complex processing much easier than in an event-based model, but tends to demand more memory, which can be prohibitive for manipulating very large documents.

XPath is a language for identifying parts of XML documents based on their elements, attributes, and location in the document structure. For example, in a MODS record we can use XPath to select the first author in a MODS bibliographic record, or all of the authors of any related items that are present in the record. XPath is most often used as a component of other XML technologies, such as XSLT.

Extensible Stylesheet Language Transformations (XSLT) allow us to transform XML documents into other XML vocabularies, HTML, or any other text representation. For example, we might transform a MODS record into MARCXML or Dublin Core for export into some other system, transform it into HTML to display to a user with a Web browser, or transform it into plain text for emailing to a user. An XSLT stylesheet is written as an XML document, and uses XPath to identify specific parts of the source document for specific transformation.

These technologies and others provide a common, system independent framework for working with, manipulating, and repurposing XML.

Why should I care?

Libraries can use XML as a common information format to exchange data with other information providers, to arrange interoperability between our own information systems, and to present information to our users. There are already a number of XML-aware systems and XML-based standards used by libraries today. This is particularly true in digital libraries. Because XML has been widely adopted for Web-based applications, a large number of tools exist to assist in building XML-based systems. The library community can share XML-based tools and standards with other industries, leveraging their experiences, and contributing ours.

XML is important to libraries as a format for manipulating information. The ability to transform XML allows it to be reused in different contexts. XML-aware systems can be made to interoperate by arranging for them to exchange XML documents. The combination of reuse and interoperability means we can use XML to knit systems together, whether to integrate different systems in our users' information landscape, or to streamline our own business processes.

XML offers flexibility in exchanging and presenting information. A benefit of descriptive markup is that data elements are explicitly marked, and knowledge of those elements allows us to transform the data as needed for exchange and presentation. Some integrated library systems use XML internally to manipulate information and format it for presentation to the user. At least one OpenURL resolver uses XML to import and export subscription data.

Exposing XML interfaces to systems can enable interoperability between library applications. This strategy is well established in many industries. For example, businesses in a supply chain may use XML to exchange purchase orders and fulfillment information in real time. The NISO Circulation Interchange Protocol (NCIP) is an XML-based protocol for exchanging library circulation information. In pilot projects, some integrated library systems are allowing remote interlibrary loan (ILL) systems to charge books out directly, greatly streamlining ILL processing. This sort of interoperability recalls the way MARC provides interoperability between cataloging systems, allowing cooperative cataloging arrangements to develop. The difference is that XML is a more flexible information container that is recognized by many more systems, and XML is often exchanged in a real-time, transaction-based environment.

Digital libraries rely heavily on XML in their production streams. The digital objects and their metadata are often created and disseminated in XML, typically using specific XML vocabularies. The display of the objects is often based on transforming that XML. Digital library standards tend to favor XML for representing metadata or digital objects themselves. This is due in part to the descriptive markup and system independent aspects of XML, and the ability to easily process the XML. An understanding of XML is critical for the planning and implementation of digital library projects.

XML is a data format that libraries have in common with the larger world. A benefit of using standards that are not confined to the library community is that we can build on the experiences, knowledge, and techniques developed outside the library community. We do not have to rely only on library vendors to develop XML tools or develop them ourselves. The time and effort saved can be applied to those activities that the standards are intended to support. This means that XML tools are tested more broadly than library-specific tools, development costs are shared among more users, and there are more tools available. To give an obvious example, there are more options for editing and processing XML records than MARC records.

All of the major programming languages, and many of the minor ones, have available various forms of support for XML. These may include packages for generating or parsing XML. There are several implementations of XSLT in different programming languages. One common processing model is for an application to receive an XML document, call an XSLT processor to do the transformation, and work with the results. Without XSLT, any such manipulations would involve programming a custom XML manipulation for each task.

XML editors assist with the creation and maintenance of valid XML documents. XML databases have been developed to allow native storage and retrieval of XML documents. This discussion only hints at the XML tools that are available.

Because XML is not a language, but a framework for developing markup languages, all of these XML tools are applicable to nearly any XML-based vocabulary.

Examples of XML-based standards

There are many XML-based standards that are used in libraries. The standards briefly described below include some of the ones more commonly encountered in the digital library community. Only a few are strictly library standards, which helps to illustrate that, as libraries, we are interacting with other players in the information landscape.

TEI

The Text Encoding Initiative (TEI) is a flexible, well-established tagset for scholarly markup, supporting a wide range of documents, including prose, poetry, plays, and dictionaries. TEI is designed to be both modular and extensible. TEI has been through several revisions. Beginning as a SGML DTD in the late 1980s, the TEI DTD was later modified to support both SGML and XML. The most recent version, TEI P5 is implemented as a RELAX NG schema. TEI is used by a number of projects focusing on online scholarly texts, such as the Brown University's Women Writers Project and the Oxford Text Archive. A partial list of TEI-based projects may be found at http://www.tei-c.org/Applications/.

The TEI began as a research effort of the Association for Computers and the Humanities, the Association for Computational Linguistics, and the Association for Literary and Linguistic Computing. It is currently being developed by the TEI Consortium. For more information on TEI, see http://www.tei-c.org/.

DocBook

Though not a library standard per se, DocBook is a freely-available, extensible tagset for technical documentation. As with TEI, DocBook began as an SGML DTD, currently supports both SGML and XML, and the next version will be expressed as a RELAX NG schema.

DocBook is used for providing much of the documentation used by library systems offices. Computer books published by O'Reilly and Associates are coded in DocBook. These source documents are used to produce both the print books and their online equivalents. [3] Sun Microsystems uses a DocBook variant for the online manual pages in Solaris. The manual you are currently reading is marked up in DocBook.

DocBook development is sponsored by the Organization for the Advancement of Structured Information Standards (OASIS). For more information, see http://www.docbook.org/ and http://www.oasis-open.org/.

EAD

Encoded Archival Description (EAD) is used for the electronic mark up of archival finding aids. EAD began as an SGML DTD, initially developed at the University of California, Berkeley, Library, and is currently an XML DTD maintained by the Library of Congress in cooperation with the Society of American Archivists. For more information, see http://www.loc.gov/ead/.

MODS and MADS

The Metadata Object Description Schema (MODS) was developed to preserve MARC-like semantics

[3] This illustrates the power of descriptive markup: one input, multiple outputs.

for bibliographic metadata in an XML format, but in a form that is somewhat simpler than MARC, uses text tags rather than numeric tags, is extensible, and is more friendly to electronic resources. Several of the examples above are based on MODS. The Metadata Authority Description Schema (MADS) is a companion to MODS for encoding authority data. The Library of Congress is the maintenance agency for both MODS and MADS.

More information about MODS and MADS can be found at http://www.loc.gov/standards/mods/ and http://www.loc.gov/standards/mads/.

METS

The Metadata Encoding and Transmission Standard (METS) schema provides a way to bundle up all of the files, metadata, and structural information for an electronic object, and draws on the experiences of the Making of America 2 project. Objects are modeled as a hierarchy of components, or nodes. Each node may have associated with it files or portions of files, descriptive metadata, administrative metadata (technical metadata, preservation metadata and rights metadata); linkages between nodes of the hierarchy may be recorded.

METS defines structural metadata and a file manifest, but relies on other schemas for descriptive and administrative metadata, which are being developed by those with the appropriate expertise. This allows METS implementers to use the best descriptive and administrative metadata standards for their projects. VRA (Visual Resources Association) Core and MODS solve different problems for different communities, and either could be used for descriptive metadata in METS. METS has become well established in the digital library world. It has been used to model books, musical scores, 45 RPM vinyl records, video segments with transcripts, and even entire websites. METS is maintained by the Library of Congress.

For more information, see http://www.loc.gov/standards/mets/.

OAI-PMH

The Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) is an XML-based protocol that allows the exchange of metadata for digital collections. Metadata from a variety of collections can be gathered, or harvested, into repositories. Services can be built on top of these repositories to enable improved access to the materials. For example, the metadata for digital collections housed at separate institutions can be gathered into a repository, and that repository might allow all of the objects from all collections to be searched as though they were one collection.

For more information, see http://www.openarchives.org/.

ONIX

The ONline Information eXchange (ONIX) is a metadata standard used in the publishing and distribution supply chain. ONIX for Books is well-established for transmitting information about books between parties in the book supply chain, and is available as either a DTD or XML Schema. ONIX for Serials is being developed as a set of XML Schemas, each focused on communicating particular information about serials. Though not a library standard, ONIX offers some interesting prospects for libraries to enhance their interactions with their suppliers. For example, Serials Release Notification (SRN) focuses on article- or issue-level information for serials. This offers future possibilities for automatically adjusting serials claiming schedules based on a publisher's most recent release information.

ONIX is maintained by EDItEUR, an international group of participants in the book and serials industries. For more information, see http://www.editeur.org/.

Summary

XML is a general-purpose information container, combining a well-defined syntax with flexibility in representation of information. XML is important to libraries for a number reasons. XML is essential to digital library literacy. The library community can share XML-based tools and standards with other industries, leveraging their experiences. XML is already an important part of our environment, and will only become more so.

Bibliography

Caplan, Priscilla. Metadata Fundamentals for all Librarians. Chicago: American Library Association, 2003.

Coombs, James H., Allen Renear, and Steven J. DeRose. “Markup Systems and the Future of Scholarly Text Processing”. *Communications of the Association for Computing Machinery* 30, no. 11 (November 1987): 933-947, http://doi.acm.org/10.1145/32206.32209.

DeRose, Steve, David Durand, Elli Mylonas, and Allen Renear. “What is Text, Really?” *Journal of Computing in Higher Education* 21, no. 3 (August 1997): 1-24, http://doi.acm.org/10.1145/264842.264843.

Ray, Erik. Learning XML. 2nd ed. Cambridge Mass.: O'Reilly, 2003.

Harold, Elliotte Rusty, and W. Scott Means. XML in a Nutshell. 3rd ed. Sebastopol, CA: O'Reilly, 2004.

TEI Consortium. A Gentle Introduction to XML. http://www.tei-c.org/Guidelines2/gentleintro.html; http://www.tei-c.org/Guidelines2/gentleintro.pdf.

World Wide Web Consortium. Extensible Markup Language (XML) 1.0. 3rd ed. http://www.w3.org/TR/2004/REC-xml-20040204/. http://www.w3.org/TR/2004/REC-xml-20040204/.

Chapter 16. What are Relational Databases, and Why Should I Care?

Vishwam Annam, Wright State University

What are relational databases, and why should I care?

Designing a database and a software application to access it is a multi-phase process. Usually, the starting phase is to architect the logical design of the database, which involves defining a database, determining tables and their fields, establishing relationships among these, identifying Primary and Foreign keys and their mappings. The next phase involves implementation of the above logical design within a specific database program using proper tools to implement data integrity. In this phase a database software (for example MySQL, Postgres, Oracle, etc.) is selected, and database, tables are created physically. In the last phase the end-user application is developed. This application allows users to interact with the data stored in the database by using interfaces (such as web pages, GUI, etc.).

This essay will concentrate mainly on the logical design of databases and provide an introduction to databases and Structured Query Language (SQL).

What is a database?

A database is a collection of related data or information, which usually consists of three elements:

- Tables
- Columns
- Rows

Tables, also referred to as entities, are the data structures holding the content of the database. For example, you might choose to store information about your library staff and the departments they work for. In this case, staff would become one table, and departments become another.

Columns, also referred to as fields, represent the attributes of a table. In the above example, a staff member can be described as an entity with its own attributes such as first name, last name, address, city, state, ZIP code, phone number, and so on, and a department can have its own attributes.

Rows, also referred to as records, represent the actual data. Whereas fields describe what data is stored, the rows of a table are where the actual data is stored. Each record is a collection of information about one specific thing -- in this example, it's your staff member or his/her department.

Types of databases

Depending on the database architecture, there are many types of databases, such as flat file databases, relational databases, hierarchical databases, network databases and more. Flat file databases, and relational databases will be discussed in this essay.

A flat file database is described as a simple database model, where all the information is stored in a plain text file, one record per line. Each record is divided into fields using delimiters (such as comma, tab, etc.) or at fixed column positions. The data is "flat", as in a sheet of paper, as compared to a more complex model such as a relational database. Strictly, a flat file database should consist of nothing but data and delimiters, with no relationships or links between records and fields except the table structure.

A single table can run into problems quickly when trying to represent anything but simple data types. While simple, this system rapidly becomes inefficient as the number of records grows, and makes it difficult for users to organize data. This system is not as secure as relational databases, and the data can be easily corrupted if more than one process is writing into database at the same time. There is no inherent locking mechanism that detects when a file is being used or modified.

Flat-file databases are ideal for small amounts of data that need to be human readable or edited by hand.

Relational databases

In an Relational Database Model (RDM), the data in different tables is mapped with relations. The goal of relational database design is to store information without unnecessary redundancy and retrieve information easily and accurately.

Unlike flat file databases, in RDM data integrity is built into the model at various levels, such as the field level, to ensure the accuracy of the data; at the table level to ensure that records are not duplicated; at the relationship level to ensure that the relationship between a pair of tables is valid; and at the business level to ensure that the data is accurate in terms of the business itself. Data is consistent and accurate due to the various levels of integrity you can impose within the database.

Relational Database Design

Data modeling

Data modeling is an act of analyzing your organization's or client's goals about the database application you will be developing. In this process, you define tables, relations and how these are related to each other. Data modeling asks the question "What?" instead of the more common data processing question, "How?"

The Components of a Data Model

1. Defining entities and their relations

The simplest data model consists of entities and relationships. As mentioned before, entities are real world objects storing data. A relationship is a significant association between two entities.

Entities can be related to each other in a variety of ways. Functional dependencies are formed when a column of one table relates to attributes of other tables. The simplest relationship is the one-to-one relationship, in which one record in a table is related to another record in a separate table. A one-to-many relationship is one in which one record in a table is related to multiple records in another table. A many-to-one relationship defines the reverse situation; more than one record in a single table relates to only one record in another table. Finally, in a many-to-many relationship, more than one record in a table relates to more than one record in another table.

2. Defining Keys

A key is an entity in a table that distinguishes one record of data from another. The key can be a single column, or it can consist of a group of columns that uniquely identifies a record. Tables can contain primary keys which differentiate records from one another. Primary keys can be an individual attribute, or a combination of attributes. Foreign keys relate tables in the database to one another. A foreign key in one table is a primary key in another. The foreign keys generally define parent-to-child relationships between tables. Database design is the process of transforming a logical data model into a physical database design and then implementing the physical model as an actual database. A logical data model is required before you can even begin to design a physical database

3. Defining Indexes:

An index is structured to make it easier to find data in the database, with fewer I/O operations. Therefore, queries can perform faster when using an index to look up data based on specific key values. Try to build indexes on large tables to support the most frequently run queries, and also create indexes on the most-referenced columns in frequently run queries in your application.

4. Views

Another aspect of database modeling is the creation of database views to support specific application data requirements. Views are not required to access a physical database, but they can be helpful to support specific application and user requirements. No physical structure is required of a view; it is a representation of data that is stored in other tables

5. Normalization

Normalization is a design approach that minimizes data redundancy and optimizes tables by systematically and properly placing data elements into the appropriate groupings. The normalized data will ensure that each entity is well formed and that each attribute is assigned to the proper entity. The normal forms are defined as follows:

First Normal Form

The objective of first normal form (1NF) is to eliminate repeating groups from an entity. When data conforms to 1NF, each attribute of the entity is a single discrete fact.

Second Normal Form

Second normal form (2NF) ensures that all the attributes of each entity are dependent on the primary key. To transform 1NF data into 2NF, create separate entities for sets of attributes that apply to multiple records and assign a foreign key to the new entity to relate it to its previous entity. Simply stated, entity occurrences should not depend on anything other than the entity's primary key.

Third Normal Form

Third normal form (3NF) ensures that no relationships between attributes exist within an entity. Every attribute in the entity should depend only on the primary key.

Further Normal Forms

Boyce Codd normal form (BCNF) is a further refinement of 3NF. Fourth normal form (4NF) states that no entity can have more than a single one-to-many relationship if the one-to-many attributes are independent of each other. Fifth normal form (5NF) specifies that every join dependency for the entity must be a consequence of its candidate keys.

6. Review data integrity

There are three types of data integrity that are implemented during the database design process.

Table-level integrity

This ensures that the field that identifies each record within the table is unique and is never missing its value. This will make certain that there are no duplicate records in a table; every record in a table is identified by a Primary key value; every Primary key value is unique; and Primary key values are not null.

Field-level integrity

This ensures that the structure of every column is well defined, that the values in each field are valid, consistent, and accurate throughout the database. Field Specifications help to warrant that the identity and purpose of each field is clear, and that all of the tables in which it appears are properly identified; field definitions are consistent throughout the database; and the values of the field are consistent and valid.

Relationship-level integrity

Also known as referential integrity, this ensures that the relationship between a pair of tables is sound and that there is synchronization between the two tables whenever data is entered, updated, or deleted.

What is SQL?

SQL, a Data Manipulation and Data Definition Language, allows users to access data in relational database management systems, such as Oracle, Sybase, Informix, Microsoft SQL Server, MySQL, and others, by allowing users to describe the data the user wishes to see.

As a Data Manipulation Language (DML), it can be used to query and update data in the database:

- SELECT - extracts data from a database table
- UPDATE - updates data in a database table
- DELETE - deletes data from a database table
- INSERT INTO - inserts new data into a database table

As a Data Definition Language (DDL), SQL permits database tables to be created or deleted. The most important DDL statements in SQL are:

- CREATE TABLE - creates a new database table
- ALTER TABLE - alters (changes) a database table
- DROP TABLE - deletes a database table
- CREATE INDEX - creates an index (search key)
- DROP INDEX - deletes an index

Conclusion

Here I only touched the surface of Relational Database Design model and SQL. I hope the information provided here can be useful as an introduction to your database application development project.

References:

Hernandez, Michael J. Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design, Second Edition

Database Types http://websiteowner.info/articles/cgi/databasetypes.asp

What is a Relational Database? http://www.wisegeek.com/what-is-a-relational-database.htm

Chapter 17. What are Indexers and Why Should I Care?

Peter Karman <karpet@peknet.com>

Introduction

Searching databases is your stock-in-trade, right? But chances are pretty good that when you enter your query into that little search box, you're not really searching the database at all, you're searching an index of the database. What's the difference, you might ask. The difference between a database and an index is not mere technical nuance; it's the difference between a book and the index at the back of the book. If you want to find some information, which method is faster and more accurate: searching the book page by page, or searching the index? For anyone designing or maintaining an information storage and retrieval system, it's an essential difference to grasp. This article will look at the differences between a database and an index; compare some of the most popular open source indexing tools; and discuss some common methodologies for designing and building an index.

Some Definitions

A database (sometimes referred to more accurately as a relational database management system (RDBMS)) is a storage and organizational system for data. It's built around the concept of the "table": columns (fields) and rows (records) layed out like a spreadsheet. Multiple tables can be related to one other through "key" fields. The result is a powerful, flexible and robust system for organizing data into conceptual chunks and then relating those chunks to one another. In the library, a database is ideal for cataloging collections: books, audiovisuals, periodicals, media of all types. A database allows you to turn data into information through the miracle of tabled organization.

An index (sometimes called an inverted index) is also a storage and organizational system for data. Unlike a database, which is organized around tables, an index is organized around something much simpler: words. An index is simply a list of words abstracted from some other data source, along with a pointer to where each word appears in the data source (a filename, a URL, a database record ID, etc.). In the case of an index at the back of a paper book, the data source is the book's text, and the book index lists words and the page numbers where they can be found. In the case of a database index, the data source is the tables, and the database index holds words and the table records in which they can be found.

Consider this example of a mailing list database:

```
+----------------------------------------------------------------+
| ID | first | last  | address          | city    | state | zip  |
+----------------------------------------------------------------+
| 1  | Joe   | Smith | 1234 Main Street | Nowhere | MO    | 12345 |
+----------------------------------------------------------------+
| 2  | Sally | Jones | 5678 Some Street | Nowhere | MO    | 12345 |
+----------------------------------------------------------------+
```

While an index of this same data might look like:

Joe, record1

```
Smith, record1
Sally, record2
Jones, record2
1234, record1
5678, record2
Main, record1
Some, record2
Street, record1, record2
Nowhere, record1, record2
MO, record1, record2
12345, record1, record2
```

A search of the index for `12345` would quickly turn up two results: `record1` and `record2`. Consider a database 10000x bigger, and you can begin to see how an index would provide a much faster search of the same data.

There are distinct advantages to using a database in the library, where vast amounts of data must be managed. But databases require that their users know how to ask for that data in ways that the database understands. The standard method of asking a database for information is known as Structured Query Language (SQL). While powerful and flexible, SQL is built around the same organizing principle as databases themselves: the table. SQL is a computer language; it is designed to tell a computer exactly what to find in the database and what to do with the results. Like most computer languages, it is exacting and precise, because like computers, it is built around a binary system.

Human language, on the other hand, is often muddy and ambiguous, because it has evolved around a different principle: the word. And so here we come to the crux: in the world of information retrieval, nothing is so intuitive and natural for human beings as the word. Thus the index is the most intuitive and natural tool for human beings who are searching for information, because like human language, the index is organized around the word.

Databases offer the power and range of a computer's ability to hold and retrieve vast quanities of organized data; indexes offer a humane way of accessing that data. Now we'll look at how the two systems can work well together.

Database and Index: Working Together

Many popular database software packages (Oracle, MySQL, etc.) offer native full-text search features. Full-text search is a fancy way of saying "search using words, not SQL". In order to make full-text search a reasonably useful (read: fast) feature, these packages often leverage an index, rather than looking at every field in every record of every table each time a search is performed. An index thus serves as a kind of "word cache" of the data in the database, in order to speed up the search.

While native full-text search is rapidly becoming standard for database software, it is often more useful and flexible to use a separate indexing system to create an independent index of your database. An independent index offers several advantages because it:

- Frees you to offer more advanced search features than the database does.
- Reduces the load on the database server. Particularly for applications where users are mostly retrieving data (rather than storing it), this can save tremendous strain on a network and server resources.
- Can be distributed amongst multiple machines. An index is a write-once/read-many kind of application, so you might create it and then copy it to different servers or sites.
- Allows you to carve out a subset of your data, or re-organize it in ways that might be more useful to some audiences or that "hide" data from certain audiences. You might, for example, create two in-

dexes from a catalog database: one with only collection titles, authors, and subject listings for the general public, and another that includes ISBN codes, inventory and circulation information for library personnel.

One convenient way of taking the data from a database and inserting it into an index is to export the database data in XML. XML (eXtensible Markup Language) is an ideal means for the interchange of data between different organizing structures because it retains the semantic meaning of groups of words in a plain text format. So a common workflow might look like:

database -> XML -> index

Many database software packages have a 'XML export' feature, and if your favorite database package does not, it is fairly simple to write a script (often in Perl) to transform the data into XML.

Once your data is in XML, the next step is feeding the XML to the indexer. There are a number of open source indexing solutions available. Some of them are known as "search engines" because they have historically been used to index web sites or collections of documents. The same principles of indexing (extract words and keep track of where they appear) apply to documents as to database records, though as we will see, the methods for retrieving and ranking the search results can differ between the two. Let's look at three popular open source indexing projects: Xapian, Lucene and Swish-e.

Overview

Lucene[1] is a Java library. The package includes example indexing and searching code, but the examples won't suffice for use with a database. You'll need to either write your own Java application using the Lucene library, or adapt an existing project to the task.[2]

Xapian[3] is a C++ library. Like Lucene, you would need to either write your own application using one of the many supported languages (C/C++, Perl, Python, PHP, Java etc.) or adapt an existing project (Omega and dbi2omega are a frequent starting point).

Swish-e[4] is a command line program written in C that includes a search library and API. Unlike Lucene and Xapian, Swish-e is a complete system: the swish-e command line tool can parse, index and search data. An API is available for writing alternate searching applications in either C, PHP or Perl.

Features

The paramount question when evaluating an indexer is: does it do what I need it to? The following table summarizes some key features of the three projects:

Table 17.1. Feature Summary

	Lucene	Swish-e	Xapian
Incremental indexing	x	x[a]	x
Synchronous search and update	x		x
UTF-8 (international languages)	x		x
Command line interface		x	

[1] http://lucene.apache.org/
[2] http://www-128.ibm.com/developerworks/library/j-lucene/
[3] http://xapian.org/
[4] http://swish-e.org/

Built-in XML parser		x	
Indexing API	x		x
Searching API	x	x	x
Stemming	x	x	x
Boolean search	x	x	x
Wildcard search	x	x	x
Phrase search	x	x	x
Proximity search	x		x
Stored document text	x	x	x
Portable index format	x	x	x
License	Apache	GPL[b]	GPL
Multiple ranking algorithms	x	x	x

[a] Experimental feature

[b] GPL plus special clause for linking against the search library

Parsing

Parsing is reading data and breaking it down into its component parts. In our case, parsing involves discerning what is a word and what is not a word. Consider this example:

```
<tag>the quick brown <foo>fox</foo> jumped.</tag>
```

An XML parser would recognize 'the,' 'quick,' 'brown,' 'fox,' and 'jumped' as words, ignoring the whitespace, the period at the end of 'jumped' and the XML tags inside the <> delimiters.

An indexer is not a parser; an indexer takes words that have already been parsed and inserts them into an index. So an indexer is dependent upon a parser to take raw data and turn it into the building blocks (words) of an index.

Incremental indexing

One of the primary features of a database is that the data is dynamic: you can add, update and delete data in the database and your changes are instantly accessible. In order for your index to keep pace with the changes in your database, the index needs to be able to increment (change) without needing to re-create the whole index from scratch. Ideally, your index should be searchable while it is being updated, so that you do not suffer from any interruption in service. Keeping the data in your database and your index in sync is very important, lest any differences cause confusion, fear or mistrust amongst your users.

UTF-8 (Unicode)

In the Good Old Days of early computing, there were only 128 text characters that a computer needed to understand. Those 128 characters were known as American Standard Code for Information Interchange (ASCII). Of course, 128 characters is about the minimum number for any one human language, and as computers increased in popularity around the world, they needed to deal with more human languages and thus more characters. So the standard character set was doubled to include 256 characters (the maximum number of mathematical possibilities in 1 byte (8 bits) of computer memory). But many human languages contain far more than 256 characters, and so the Great Brains of the computer world finally came up with a system that allows for 1000s of characters. That system is known as Unicode. UTF-8 (short for Unicode Transformation Format -- the 8 is an allusion to its byte orientation) is one very common implementation of Unicode, and is in fact the default character set for XML.

Modern computer programs (meaning they have been written since the mid-1990s) usually support

UTF-8, though exceptions of course exist. If the data in your database uses UTF-8 (or any other Unicode character set, sometimes called "multibyte encodings") then you will need to use an indexer that supports Unicode.

Searching

Once you've got your index created, you'll want to search it, of course. Experience with internet search engines tells us that most folks enter just one or two words when they search. [5]But a good search tool allows for more complicated queries. Here are some common features:

Stemming

If a user enters the word `run` into a search box, should your search tool return matches for the words `running` and `runner` as well? If you think those should count as valid matches, then you need stemming. Stemming means reducing a word to its root base. There are several models for deducing the stem of a word, and many indexing tools contain different models to choose from.

Stemming has other benefits: it will reduce the size of your index because fewer unique words will be added to the index, and it can increase the chances of finding just the right thing when you aren't sure of the exact form of a word. On the other hand, if you only want matches to the word 'run' then stemming will actually be a detriment, since you'll get false positives amongst your matching set. See the Methodologies section at the end of this article for more approaches to stemming.

Boolean

Boolean is a fancy word for joining your search terms with logical operators like AND, OR and NOT. For example, to find all the records that include both the word `dog` and the word `cat`, you might search for `dog AND cat`. To find all the records that include either of the words, you might search for `dog OR cat`. OR searches, as a rule, will result in a bigger set of matches than AND searches. NOT is a way of omitting records that match a word. A search for `dog NOT cat` would find only records that contain the word `dog` but not the word `cat`.

Wildcard

Wildcards allow you to guess at the spelling of a word by using a special character to represent any character. For example, a query like `car*` will match all records that contain the words `car`, `cartesian`, and `cart`. The * character is the wildcard character that means "match zero or more characters of any kind." Similarly, the ? character might mean "match exactly one character of any kind." Wildcards allow you more flexibility in your searches.

Note

Wildcards are not the same thing as stemming, even though they seem to behave the same way. Stemming has to do with the semantic root of a word, while wildcards have to do with matching patterns of characters. The stemmed root of `runner` is `run`, while a search for `runn*` will only find `runner`, not `run`.

Phrases

A phrase is a group of two or more words in the exact order and position they appear in the record. Usually a phrase is indicated through the use of "double quotation marks." For example, a search for `"fast runner"` would match records that have the words `fast` and `runner` adjacent, in that order. It would not match records that contain both words in a different order (like "`runner fast`") or in a different location ("`fast is the sparrow, slow is the runner`").

Proximity

Proximity searching is similar to phrase searching, only with greater freedom with respect to word order and location. In other words, you can search for words that are near one another, but not adjacent. For example, a search for `runner NEAR fast` might find records with `runner fast`, `fast runner`, and `fast like a runner`, but

[5] http://www.niallkennedy.com/blog/archives/2005/04/recent_innovati.html

not records where `fast` and `runner` do not appear near one another. "Near" can be defined as a range, as in "within 10 words of one another." The proximity range should be configurable by the user, as in `runner NEAR:10 fast`.

Ranking

Ranking is one of the most important search features an index can offer. Ranking is a mathematical way of measuring the assumed relevance of a set of matches. Relevance can be measured in lots of different ways, and for any given user and query, one relevance method might be better than another. The best search tools allow for ranking to be adjusted or configured by the user, so that the user (who is ultimately the best judge of relevance) can tweak her search results to help her find the records she's looking for more quickly.

One common ranking method is called Inverse Document Frequency. It uses a variety of mathematical algorithms that boil down to this: the more frequently a word appears in a given record, the more likely it is that record is relevant to your search. However, if the word appears frequently in many records in the index, then that word is likely not a good word for measuring relevance. For example, if you search for `the fast runner`, the word `the` probably appears too many times in the index as a whole and should be ignored when determining relevance. The word `fast` might appear often, but not as often as `runner`, so `runner` is the most important word in the query. The ranking algorithm takes all that into account when sorting the matches and presenting them to the user.

Summary

Lucene

Pros

- most flexible search syntax (wildcards, ranking, etc.)
- fastest search for complex queries
- ports to other languages (Clucene in C++, Plucene in Perl, etc.) shows development interest

Cons

- little out-of-the-box support
- requires additional (Java) programming to integrate with database

Xapian

Pros

- probablistic ranking
- best scaling (index size)
- most extensive API support

Cons

- poor documentation
- requires additional programming to integrate with database

Swish-e

Pros

- best out-of-the-box support
- fastest search for simple queries

- good documentation and examples

Cons

- no UTF-8 (international language) support
- no stable incremental indexing (ability to add/update/delete records)
- limited search syntax
- simplistic ranking
- no indexing API/library

Which tool you choose depends on a lot of factors, among them: the size and type of data you intend to index; the technical resources you have available in setting up your system; and what kind of query flexibility your users will require.

Swish-e is the easiest to use and configure; allows the most flexibility with respect to input and defining what a 'word' is; and is fastest for most queries. However, Swish-e lacks UTF-8 support, incremental indexing, and does not scale particularly well for large (>million) data sets.

Lucene and Xapian offer great flexibility with indexing and searching, and can handle large data sets. However, neither of them provide much out-of-the-box support, nor do either of them parse your incoming database data, leaving that task to some other application.

Methodologies

Like human languages, indexes suffer from some limitations when they meet the cruel binary world of the computer. This section suggests some tips to keep in mind when you create an index.

What's in a word?

What combination of characters constitutes a "word" in your data set? That might seem like an obvious question, but it can have a significant impact on the usability of your search tool. Consider this sentence:

If you click on the link marked http://foobar.com/ it doesn't work.

Which series of characters should be considered words, and which should not? Splitting up the sentence on whitespace is a start. If all non-whitespace characters are considered valid word characters, then you would end up with a word list like:

If
you
click
on
the
link
marked
http://foobar.com/
it
doesn't
work.

But do you really want `work.` treated as a word? Shouldn't it be `work` without the trailing period? Otherwise, someone searching for `work` won't find that sentence. But that's trivial. Consider this sentence:

If you click, beware: it doesn't work; instead, a message like 'help!' appears.

While that's a perfectly valid (albeit awkward) sentence, splitting the characters up on whitespace alone leaves you with:

If
you
click,
beware:
it
doesn't
work;
instead,
a
message
like
'help!'
appears.

Ah! You say. Just ignore all the punctuation marks: `,;!.?'` But is it that simple? Look again at the first example. The characters `http://foobar.com/` would get split up into an awkward series of words like:

http
foobar
com

which isn't ideal either, especially when someone wants to find the URL `foobar.com`, as opposed to `foobar.org` or `foobar.edu`. A similar false splitting would happen to the word `doesn't` and all contractions like it.

Obviously, what constitutes a "word" isn't as easy as it first appears. Some search tools get around the awkward splitting of words by making everything a "phrase" search by default. Depending on your data set, you may need more control over the technical definition of a "word" than some indexing tools will offer.

Tip

If your indexer supports the definition of word characters, configure it to ignore punctuation only when it appears at the beginning or end of a string of non-whitespace characters. That way `doesn't` gets indexed as `doesn't` but `'help!'` gets indexed as `help`.

To stem or not to stem

Stemming is a powerful feature that helps reduce the size of your index and helps users find related information more quickly. But stemming is also imprecise and fuzzy, and can lead to lots of "false positives" when you're looking for specific information. Stemming can sometimes hurt, more than help, the effectiveness of your search.

Tip

To increase the usability of your search system, consider creating multiple indexes: one with stemming on and another with stemming off. Then create a search interface that offers your users a choice between stemming and no stemming. While it will likely more than double the size of your filesystem requirements, hard drive space is cheap these days. And offering your users a choice when searching is well worth it.

Hide and seek

Full text search is attractive because it mirrors the way humans think: in terms of words. But as everyone who has ever used a search tool knows, figuring out which words to use in a query can sometimes be confusing (at best) and infuriating (at worst).

When you create your indexes, you have the opportunity to make the search process easier for your users by manipulating the data you put into the index. Stemming is one common approach; there are many others. Consider misspellings, for example. It's common for users to misspell query words, resulting in no matches when, if they spelled their words correctly, there are many matches to be found. Synonyms are another example: a user enters `run` and misses all the records that mention `melt`, `flow`, `tend`, `incline`, etc.

Tip

Just as in Stemming Tip, create multiple indexes. Include the literal data, common misspellings, synonyms, related words, etc., in one index, and just the literal data in another index.

Feedback

No indexer or search tool is perfect. One of the hottest research topics in current computer science is how to more accurately store and retrieve vast quantities of diverse information. Ranking is a huge issue, especially in the area of e-commerce, where the top ranked sites in search engines like Google and Yahoo! are getting the most dollars flowing their way.

Every audience is different, and uses search engines for different reasons. But patterns emerge. Discovering what the search patterns are amongst your user groups can make a dramatic difference in improving the quality of their search experience.

Tip

Every decent search tool offers logging of user queries, whether in a web server log or similar tool. Study how your users search, try some usability studies, and accomodate the patterns you discover by creating multiple indexes, offering search tips, accounting for commonly misspelled words, adding additional data to your indexes, and making recommendations based on previous searches. And most important: listen to your users. There are patterns lurking in their comments and questions.

Conclusion

A good indexer and search tool can help tame your databases and bring the data closer to your users. It takes practice, experimentation, and quality feedback to help tune your indexing methods to better serve your needs. Be patient. Be persistent. Join search tool e-mail lists and ask questions. It's likely that someone has run into a situation similar to yours before. And if at all possible, have fun.

Part VII. Implementation and maintenance

Table of Contents

Chapter 18. Implementation and Maintenance

Eric Lease Morgan, University of Notre Dame

This essay outlines one way to implement and maintain a MyLibrary instance. The following process is suggested:

1. Assemble a team of people to do the work.
2. Give the team the necessary resources to accomplish the job.
3. Answer questions regarding information architecture.
4. Install and configure MyLibrary.
5. Fill MyLibrary with content -- do input.
6. Create interfaces to provide access to the content -- facilitate output.
7. Do usability testing against the interfaces.
8. Maintain the content.
9. Evaluate and go to Step #1.

Each of these steps is described in more detail in the following sections.

Assemble a team of people to do the work

You will need to assemble a team of people to do the work, unless of course Leonardo Da Vinci works in your library. Few people possess all of the necessary skills. At the very least your team will probably consist of a:

- systems administrator
- Perl programmer
- graphic designer
- subject specialist

The systems administrator is responsible for maintaining your computer's hardware, software, and networking infrastructure. They need to be knowledgeable about operating systems, filesystems, users/groups, and Internet connections. They are the people who install and configure things like Apache, Perl, and MySQL. Some places have computer centers who routinely do these sorts of activities. Working with the programmer, the systems administrator will install the MyLibrary Perl modules. Once MyLibrary is installed, the systems administrator will be primarily responsible for making sure the computer is running smoothly. Make sure they back up your data on a regular basis.

The Perl programmer is responsible for creating functional interfaces to the underlying MyLibrary database. Some of these interfaces are computer-to-computer interfaces such as the importing of MARC records from a catalog or the exporting of Real Simple Syndication (RSS) feeds. Other interfaces will have human components, and in such cases the programmer will need to work closely with the graphic designer. The programmer is not expected to create everything from scratch since the MyLibrary distribution comes with a number of sample interfaces. You might want to simply use one of these interfaces instead of creating your own. It is essential for the programmer to be familiar with object-oriented programming techniques and common gateway interface (CGI) scripting.

The graphic designer is responsible for making sure your human-to-computer interfaces are usable and aesthetically pleasing. (Usability is different from functionality.) They need to have an in-depth knowledge of HTML, XML, cascading stylesheets, and the principles of user-centered design. Ideally the user interfaces written by the Perl programmer will output rudimentary HTML with plenty of HTML class and id attributes to be used as hooks for the cascading stylesheets. Through the stylesheets the graphic designer should be able to modify the look and feel of the interface. This is called separating presentation from content. The graphic designer should also be an advocate for usability testing, and they should have a thick skin enabling them to take criticism well.

Finally, the team will require someone who is knowledgeable about content, a subject specialist. This person will bring to the team the principles collection development, cataloging and classification, as well as reference services -- all of the traditional activities of librarianship. This person will be the primary driver behind the process of answering questions regarding information architecture, outlined below. Once the questions are answered, the subject specialist will be responsible for putting the answers into practice through data entry. The subject specialist will need to articulate sets of facets and terms, select information resources, and enter everything into the system accordingly. The subject specialist should also be keenly aware of user centered design principles because the nature of librarianship has dramatically changed with the advent of the Internet. Expectations regarding the access and use of information now are quite different from the expectations of ten years ago.

None of the people and skills outlined above is more important than the other. Each is equally necessary for a successful implementation. At the same time you might consider supplementing your team with people with more specialized skills such as:

- relational database design and implementation
- indexing techniques
- advanced XML applications and XSLT programming
- conducting surveys and doing statistical analysis
- facilitating focus group interviews and usability studies
- creating and maintaining controlled vocabularies
- doing large volumes of data entry and maintenance

Each of the activities and skill sets is described in greater detail throughout the book. You are encouraged to consult those chapters for more detail.

Give the team the necessary resources to accomplish the job

Computer hardware/software and time are the necessary resources for the team to complete their implementation.

The hardware/software requirements for implementing MyLibrary are minimal. Really. About any Intel based computer with at least 512 MB of RAM and 2 GB of disk space will do just fine in terms of hardware. The more RAM the better. Now-a-days it is uncommon to have a computer with less then 20 GB of disk space. If you were to purchase a new computer to host just MyLibrary, then $2,000 will buy you a great piece of hardware.

MyLibrary is essentially an open source, LAMP (Linux, Apache, MySQL, Perl) system. Therefore, if you or your institution already have a Linux computer up and running, then it will probably work quite well. MyLibrary is designed to do input/output against a relational database. The MyLibrary installation process is designed for use the relational database program called MySQL. Since MyLibrary is a set of Perl modules, you will need Perl installed on your computer. Any version of Perl 5.0 or later will work. We use version 5.8.5. The MyLibrary modules require a number of other Perl modules. The easiest way to install these modules is through the use of CPAN. Invariably you will want to serve your MyLibrary content over the Web through an HTTP server. We use Apache. Any HTTP server will do as long as you are able to run CGI scripts from within its filesystem. Your systems administrator and Perl programmer are expected to understand these details.

In short, if you wanted to start from scratch you could probably use one of the desktop hand-me-downs lying around the office. Install on it Linux, Apache, MySQL, Perl, and you are ready to go. If you already have a computer in place, and it already has Apache, MySQL, and Perl installed, then that computer will work just fine too.

Time is by far the more expensive resource necessary to fully implement MyLibrary. Time will need to be allocated in a number of ways. First of all, time will need to be spent allowing the team members to actually become a team. Many people think this process is too "touchy-feely". On the other hand, the sooner the team establishes norms of behavior, decides how to build consensus, and learns how to work with each other the more quickly your implementation will come to fruition. This is especially true if the team members do not regularly work together. Allow the members to go on a field-trip or two as well as one more more retreats. Feeding them helps too.

Second, time will need to be spent answering the questions of information architecture. On the surface this too appears to be a lot of "navel gazing" but time spent addressing these issues will uncover hidden assumptions, help you set priorities, outline the problems MyLibrary will be expected to address, and build relationships with your patrons. While this work does not produce a whole lot of tangible results, the result forms the entire foundation of your implementation.

Finally, time will need to be spent doing the work normally associated with the implementation of computer technology. Setting up hardware and software. Writing and/or configuring computer programs. Customizing interfaces to meet your specific needs. Filling the system with data. Maintaining the data. Evaluating success. Repeating the entire process. Here again, remember that any computer implementation consists of 20% computer work and 80% people work.

Using the interfaces supplied with the MyLibrary distribution, a competent Perl computer programmer should be able to install, configure, and make accessible a couple dozen Internet resources through a searchable/browsable interface in about two days. This would be a good introduction to the MyLibrary application, however it wouldn't address issues such as implementing a robust hardware/software infrastructure, integrating the user interface with the balance of your site, or applying the principles of information architecture.

If you wanted MyLibrary to be the primary driver of your library's website, then the entire implementation process might take as long as a year. The time you spend will not necessarily be computer-related but related to the whys and wherefores of the system as well as ongoing maintenance.

Answer questions regarding information architecture

The first thing for the implementation team to do is to answer questions regarding information architecture. There are essentially three questions to be addressed:

1. Who is your audience and what are their needs/desires?
2. What is the purpose of your implementation and how does it fit within the context of your institution?
3. What type of content will your implementation contain, and how will it be conceptually organized?

These questions were elaborated upon in a previous chapter.

As answers regarding information architecture are articulated, write them down and share them with the stakeholders throughout your institution -- both inside and outside the library. Answering these questions is a never-ending process. Regularly revisit the answers regarding your information architecture.

Install, configure, and fill MyLibrary

Technically speaking, MyLibrary is a set of object-oriented Perl modules providing the means for doing input and output against a specifically shaped relational database. Therefore you will need a computer with Perl installed with hooks to a relational database. MyLibrary is presently configured to use MySQL as the database, but without too much tweaking it should be able to do input/output against other relational databases such as Oracle or Postgres. Similarly, MyLibrary was developed on top of a Unix operating system, but people have installed it on the Windows platform.

Assuming you already have MySQL installed, below is an outline of the necessary steps used to install the Perl modules. Much of this process is done for you by running the perl Makefile.PL command from within an uncompressed MyLibrary distribution:

1. Create a MyLibrary database using the sample-data.sql or STRUCTURE-ONLY.sql files found in the distribution's db directory.
2. Create and configure a MySQL user with permissions to read and write to the newly created database.
3. Edit MyLibrary's Config.pm module to record the network location of the database as well as the username/password of the authorized user.
4. Install the Perl modules.

Once this is done you should be able to write CGI or command-line driven scripts allowing you to do various types of input and output against the database. Many sample scripts are located in the distribution's bin, cgi-bin, and cgi-admin directories.

Almost invariably you will want to use a Web-based interface to do at least some of your data-entry. The cgi-admin directory contains a family of CGI scripts allowing you to do this. Like all the other scripts in the distribution, the scripts are only samples. Save them in a directory on your Web server where CGI script execution is permitted and begin data-entry. To do so try this:

1. Articulate a set of facet and terms used to provide the conceptual organization of your content. This is described in more detail elsewhere in this document.
2. Use the administrative interface to enter the facets and terms.
3. Optionally, add descriptions of one or more librarians and be sure to associate them with one or more facet/term combinations.
4. Create at least one location type. Information resources take many forms as do their location types. For right now, create a location type called something like "Internet resource". These location types will take the form of URL's.
5. Finally, use the administrative interface to add resources to your collection. At the very least you will want to give each resource a title, a description, and a location (URL). You will also need to associate each resource with at least one facet/term combination.

Since MyLibrary is really a set of Perl modules and not an application, data-entry can be done from the command-line as well as in batch mode. For example, here at Notre Dame we regularly dump sets of MARC records (supplemented with facet/term combinations) from our catalog, convert these files into RDF/XML files, and import them into our MyLibrary database. Alternatively, since the fields in the underlying MyLibrary database are a superset of the basic Dublin Core elements, it is possible to harvest content from OAI-PMH repositories and cache it to MyLibrary. This provides another way to fill a MyLibrary instance.

See the section called *Example applications* for more details.

Create interfaces, and do usability testing

The MyLibrary distribution includes a few sample interfaces to your MyLibrary implementation. These interfaces will not, nor are they expected to, satisfy the needs of every institution. Instead, they are examples of how the underlying system can be exploited to meet your and your patron's needs. For example, here is a list of some of those interfaces:

- mylibrary2oai.pl - make your MyLibrary content accessible via OAI-PMH
- mylibrary2rss.pl - make your MyLibrary content accessible via RSS feeds
- mylibrary2sru.cgi - make your MyLibrary content accessible via search
- mylibrary.cgi - make your MyLibrary content accessible via browse

Each of these sample applications are described in more detail later in this book. Programmers are expected to read the Perl API, examine the code from the sample applications, work with the balance of the MyLibrary team, and write programs fitting your particular needs.

Usability testing is a highly structured communications process. It is not science. The word "test" is a misnomer. A better word might be "study". Other sections of this book describe usability studies in much greater detail.

Finally, and just as importantly, make sincere efforts to practice user-centered design when creating your interfaces and doing your usability testing. The Internet has significantly changed users' expectations regarding the access and use of information. The older roles of libraries learned in library schools are increasingly outmoded. We librarians need to rethink much of what it means to be a librarian in an era of globally networked information. Put less emphasis on personal experience and anecdotal evidence. Instead, use focus groups, surveys, log file analysis, and usability "studies" to form the basis of your decision-making.

Maintain content

Once you have a production implementation of MyLibrary in place the largest ongoing activity will be maintaining the content. How you do this depends a great deal on the types of content in your implementation, where it originates, and where it is used.

For example, if your content primarily comes from your catalog and gets imported into MyLibrary via sets of MARC records, then maintaining your content will be a matter of maintaining your catalog. You already have processes in place for this type of work.

If your content comes from OAI data repositories, then maintence will most likely take the form of regularly running programs against those repositories.

More than likely, your content will be a mixture of things from your catalog and sets of Internet resources usually not deemed worthy of putting in your catalog. (For example, items you do not own or license.) In these cases you will probably use a combination of automated and manual data-entry methods. The records in your system that were entered manually will need to be regularly examined. Do the links still work? Are they still relevant according to your overall information architecture? Do they still fit within your collection development policy? If not, then you will need to update or weed them from your collection.

The organization of MyLibrary content is postulated on sets of locally-designed facets and terms -- a controlled vocabulary. By definition, a controlled vocabulary is a form of human language. Language is ambiguous and ever-changing. It will be necessary to monitor your facets and terms, updating them as time goes on. Do you need to create new subject facets? Have new audiences become a part of your community and will it then be necessary to create an audience facet? Do you now have access to new types of information like sounds or data sets? If so, then you may need to update your facets and terms. Does your hosting institution (college, university, company, or municipality) host a portal? Do you want to advertise not only your information resources but also your services in the portal? If so, then you may need to go beyond the traditional facets such as subjects, formats, and research tools, and enhance the them with things like help and bibliographic instruction.

Processes for maintaining your content will differ greatly from library to library. Consider reallocating existing personnel for the task. In principle the maintenance process is similar to the maintenance process of other content in your library. The difference is only the environment in which it takes place.

Evaluate and repeat

Library work is never done. Students come and students go. Younger people get older and require/desire different aspects of library service. Collections are rarely complete. Technology is constantly changing, and these changes modify user expectations. Priorities are modified over time. Budgets fluctuate.

For all these reasons it is a bad idea to think of your MyLibrary implementation as a static thing. It will need constant monitoring. Is it getting used in the manner you expected? It it meeting expressed user needs and desires? Does it cost more than the perceived benefits? On a regular basis you will want to ask yourself these sorts of questions, and depending on the answers you will want to return to Step #1.

Good luck.

-- Eric

Part VIII. Appendices

Table of Contents

Chapter 19. Pioneering Portals: A History Of MyLibrary@NCState

Keith Morgan, North Carolina State University Libraries

Introduction

As the twentieth century came to a close the phenomenal growth of information technology applications seemed limitless. University libraries, like all sectors of the new information economy, experienced tremendous growth in the number of scholarly digital resources that could be made available to students, faculty, and staff of their research communities. As late as 1994, a well-funded research library of average size might have had no more than four or five electronic bibliographic databases that could be directly used by the campus community. There were still vast realms of information available through such information providers as Dialog, Dow-Jones News Retrieval, and Lexis-Nexis but these services were expensive and generally required the mediation of an expert searcher. These new tools, often CD-ROM databases networked through a "juke-box" application, were the first that searchers could use directly with little intervention from library reference desk staff. Some training or other basic orientation was often needed but once some familiarity was attained the basic information-seeking dynamic began to change.

Despite these foundational changes, few librarians, even those intimately involved with developing information technology, could have foreseen the tremendous changes that the growth of the Internet and the success of the World Wide Web browser would bring to all libraries, not just those in the university research community. The moderately well funded university library of 1994 with four or five bibliographic databases was, by 1997, offering a hundred or more bibliographic, and occasionally even full-text, databases. Added to this mix were a growing number of full-text electronic journals. Some journals were new scholarly publications created to take advantage of the new digital publishing opportunities and others were well-established print journals now also offering their contents in electronic form. Libraries were also forced to take notice of the ever-growing world of "Internet resources." Many professional organizations and even individuals created web sites with pages of information that had value for scholarly research. These sites, especially in the early days of the web, were often free of charge and libraries were faced with a new conundrum of inventing ways to point students and faculty to valuable, but off-site, resources. Organizations and individuals creating web sites that were erroneous and prejudicial to the research process compounded this problem.

Another aspect of the social and technological problem facing libraries was the growth of network access. As the ability to dial in to university computer networks expanded, more students and faculty members were able to conduct some aspects of their research from home. The traditional ability of reference librarians to answer questions and assist in the research process was being eroded by this new mode of network research. It is the influence of this new environment that stimulated university libraries to begin a critical examination of the idea and purpose of the library web page.

As the number of resources that could be delivered from a library web page grew, libraries were forced to address the question of defining the purpose of a web presence. Many library web sites grew quickly as new sources of digital information became available. But as screen real estate was limited, the question of what types of information had priority was also debated. As more bibliographic and full-text information was added to library web pages, it became apparent that the library web page was now the essential starting point for beginning any research project. Instead of using a web page to point out the resources, the web page was used to directly access the resources.

The typical university library web page in 1996 reflects how libraries first began to use the Web to arrange networked information. Through the work of the Internet Archive Project, which collects and saves web pages, it is possible to inspect examples of how university research libraries first conceived of a web presence. Libraries were pointing to information about the libraries, as well as a small selection of library databases, catalogs, and other resources. The information was presented in a linear fashion -- the ubiquitous images, frames, and mouse-overs of today are absent. Yet common to all is an attempt to select the most relevant information for a diverse community and present it in this new medium.

The Emergence Of The Gateway

As the implications of this new and emerging environment began to be more fully understood, many libraries looked to the dynamics of the emerging commercial Internet for ideas. There were two commercial applications that caught the attention of innovative university libraries. The first Internet trend to affect the organizational structure of the library web was the concept of a gateway. The reasoning behind a gateway was that if an Internet host was to centrally locate a selection of the most popular web sites on its home page, they would draw regular users back to their site. At this time the possibility of generating revenue was not clearly understood and the idea of generating repeat visits from a loyal group seemed to be a good method of guaranteeing traffic flows. These first commercial gateways would gather together a collection of general-purpose Internet resources. These would often include a newswire, a link to the National Weather Service, sports scores; movie reviews, horoscopes and financial information such as a time-delayed stock quote service. Such collections could be as broad or as deep as the gateway designers wished to include.

The Challenge of the Portal

As more and more web sites offered some degree of personalization and customization options, the idea of the gateway began to be challenged by the idea of the portal. In the beginning many people considered a portal as a term synonymous with gateway, an anchor site or major starting point. As the implications of portals with personalization and customization options began to disseminate in the Internet community, innovative libraries saw this as a second opportunity to control the growing number of available scholarly resources while, at the same time, providing patrons with a measure of personal control. Portals and gateways began to evolve in somewhat different directions.

One advantage that a personalized library portal has over a gateway is that by allowing patrons to identify their research interests a library can "push" a selection of recommended databases, electronic journals, Internet resources, journal table of contents, and new monographs to those who identify with a particular discipline. Internet history is often vague but three of the first university libraries to provide portals were Cornell, North Carolina State, and the University of Washington. For Cornell and Washington the portal was an outgrowth of earlier work developing a library gateway. At NCSU the goal was to discover a method of improving on an ever-expanding web site.

Genesis

In early summer of 1997 the newly formed Digital Library Initiatives Department (DLI) of the NCSU Libraries was charged with the development and administration of the annual user survey. For 1997/98, the area of electronic resources and services was selected as the second topic in a systematic program of user surveys. Over the last decade, library expenditures for electronic resources and computing equipment had increased dramatically, and the library had introduced new services and staff to serve users' information needs. As new information technologies permeated teaching and learning activities, library resources were stretched thin in an effort to meet the emerging demands of the digital environment while continuing to support user needs for print collections.

With the help of tuition increase monies starting in 1996/97, the NCSU Libraries was able to make a commitment to expand its collections further and to establish the "digital library." Library administration recognized the importance of ensuring that library expenditures truly address user needs and of determining those needs in the rapidly changing digital environment.

While the general topic of this survey was digital resources, the specific mission was to determine what users required in the area of digital resources and services, how well the Libraries were meeting their needs, and what might be changed to better meet those needs. The survey took the form of focus group discussions, the results of which were equally distributed between comments expected and unexpected. One interesting discovery from the focus groups was that many members of the university community, while happy with the digital largesse manifested in the increasing number of databases and electronic journals, were also concerned about their ability to assimilate and manage this electronic bounty. Many participants in the focus groups looked to the Libraries to provide guidance in managing an environment that was rapidly threatening to become information overload.

What was to become MyLibrary@NCState had its genesis in conversations regarding a possible application to the spring 1998 National Science Foundation digital library grant program. During the first meeting of the DLI, on January 16, 1998, the idea of applying autonomous agents and intelligent profiling to build a personal library information manager was discussed. Such a tool could allow members of the campus community to individually define for themselves which of the many available library resources were most important to them. At the time of this meeting new Internet personalization services such as MyYahoo or MyExcite had recently become available. It was clear the digital resources and services of the Libraries were obvious candidates for personalization and customization options. Other research projects such as Firefly and Leitizia at MIT and Informant at Dartmouth also suggested relevant design considerations for a similar model.

During this initial meeting two essential factors were considered. First, what was the most important issue that such a project could address? Brainstorming this idea resulted in an agreement that the Libraries should explore the idea of how to develop the capability to organize, assemble, and present to the user a "customized library" reflecting his/her major interests and needs, an attractive and intuitive listing of or gateway to multi-format information resources and services. This "virtual" library environment could be further customized and enhanced over time in order to link to search results or personal references, add/remove/change appearance of resources, or document successful search strategies.

The second factor discussed was what might be the "core innovative idea" that distinguished this project from others, current and previous. Meeting participants concluded that this core idea should focus on enabling the library user or information seeker from their perspective; moving beyond a primarily "computing-oriented" approach to defining and demonstrating the concept of a digital library. Building a digital library inhabited by people: recognizing the "human element" and bringing users and librarians closer together through common conceptual models and the use of technology for real-time collaboration/consultation.

Although it was decided not to pursue the NSF grant at that time, DLI took those initial ideas and in a short span of time developed a basic model for such a service. Not all of the ideas generated at these initial meetings could be initially incorporated but a foundation for further development was established. All facets of this first model were examined and a number of features and improvements were integrated. Once the working model was completed, DLI made appointments with a number of librarians from various departments to inspect the model and comment on the idea. Nineteen librarians, including five department heads, were shown the model in a span of about three weeks and their comments and suggestions formed the basis for an improved model.

Determining Relevance

The MyLibrary model was an attempt to provide a solution to the critical problem of information overload. Although the model presented a small step in technology development it was intended to offer a quantum leap in service and institutional recognition. The library at NC State, like other university research libraries, is the central storehouse of information, and user surveys at that time showed an unceasing demand from users for more, not less, information. However, the more databases licensed, the more monographs purchased, and the more serial runs increase, the more incumbent it becomes to provide information access shortcuts and strategies. User surveys and focus groups underscore this need by revealing a growing frustration with access to information at the same time that more information is requested and needed.

Ramifications

Initial discussions conducted by DLI with library staff found a clear recognition that the MyLibrary model deserved further exploration. However, there were significant questions to answer, including scalability, profile enhancements, improved user interface, administration and maintenance, infrastructure integrity, documentation, and other issues. The immediate challenge was to form a project team that could build on and improve the model as it now stood. It was clearly recognized that this model, and the service it would generate, had the potential to impact the work of nearly all library departments, and that representation at design phase was essential to success. As a result the MyLibrary Development Team was formed in late 1998 to refine and expand the initial project model. This team included representatives from other departments within the Libraries, ensuring that other constituencies had an immediate voice in project development.

System Components

The essential idea behind MyLibrary@NCState was that using any Web browser, NC State faculty, students, and staff could create a personalized profile based on specific academic disciplines. After the subscriber made a series of discipline-related selections, MyLibrary@NCState built a customizable interface to the library's resources for the subscriber. These resources included links to electronic journals, citation databases, direct access to common search engines, and discipline-specific Internet resources. After the initial page was generated, other links could be added to or subtracted from the preconfigured list of recommendations to create an individualized "digital library." For example, a subscriber might select among the system's many "customize" hotlinks to display the list of disciplines. Each discipline list is associated with information resources (journals, databases, Internet resources, etc.) specific to that field. Resources from any discipline could be chosen for inclusion in the subscriber's personalized digital library, which appears in a Web browser after the selections are submitted. The next time a subscriber visited MyLibrary@NCState, the system remembered all customizations and listed them accordingly.

MyLibrary@NCState had four features that made it much more than a simple bookmark manager. Current Awareness was a service allowing subscribers to receive and search on a regular basis lists of new books that had been added to the library's collection. Using Library of Congress classification numbers subscribers could create any number of current awareness profiles. The MyLibrary@NCState system saved these profiles and regularly searched its database for them. Search results were sent to a subscriber's e-mail address, clickable URLs within the message allowed direct access to the library's catalog and to more information describing the book.

Based on the selected discipline, the system also displayed the name and contact information for the appropriate collection manager and reference librarians who specialize in that subject area. More than one librarian could be listed, depending on the disciplines selected. From the focus groups it was learned that, although digital libraries are desirable, users want direct access to librarians as well.

Students and faculty or administrative staff do not always have time to search for new and better information resources. MyLibrary@NCState assisted in this area with another section called "Message from the Librarian." This section--updated regularly by the appropriate librarians--announced, suggested, and helped subscribers keep abreast of interesting information developments in their selected disciplines.

Finally, unlike bookmark managers, MyLibrary@NCState was portable and required only a Web browser and Internet access. Because admittance to the system's database is keyed to NC State's registered user identification accounts, valid subscribers can reach MyLibrary@NCState from nearly any Web browser in the world, whether their computers are in offices, homes, or even the library. Conversely, bookmark managers or locally defined sets of bookmarks are bound to a particular machine or computing system.

Policy, Content, and Technology

In June of 1999, the Director's Council of the NCSU Libraries created a standing management committee to guide MyLibrary@NCState from its development phase into life as a production service. Provision for knowledge transfer between the project development team and this new committee took the form of two members of the former serving in an ex officio capacity on the latter and a chairperson intimate with the details of MyLibrary@NCState's nativity and infancy. The roster of departments represented on the management committee included Systems, Cataloging, Collection Management, Research and Information Services (RISD), and Digital Library Initiatives.

A charge statement was provided to the management committee, a map to guide its weekly meetings. This statement comprised eight mandates:

1. Define the role of staff in maintenance of the product;
2. Integrate necessary procedures into departmental workflows;
3. Develop a marketing strategy for the release of future upgrades;
4. Develop a training manual for staff;
5. Further refine the product's discipline and content structure;
6. Recommend additions and enhancements to the product for future release;
7. Revise and recommend upgrades to the administrative pages;
8. Ensure the quality and integrity of the product as well as its continuation as a vital service to the NC State community.

Open Source

Following the release of the first version of MyLibrary@NCState the concept generated an ever-increasing amount of interest at both the campus, national, and international level. In 1999 MyLibrary@NCState was named one of the top technology trends by the Library and Information Technology Association and presentations by DLI staff at CNI, the LITA National Forum, and ASIS spawned more interest. From the beginning of the design process DLI staff held to the idea that it would best for this service if it could be shared with other university libraries. Indeed, every presentation generated queries; both in person and in follow up e-mail, as to whether DLI would be sharing the code. As these queries were examined, a plan began to be formulated that would allow the enhancement of MyLibrary@NCState development in a collaborative environment while at the same time sharing the code with those libraries interested in having it.

On February 18, 2000 e-mail messages were sent to all of the people who had inquired as to obtaining the source code. This was followed by a second general notice to a number of listservs. In less than two months the source code was downloaded approximately 300 times, 106 people subscribed to the mailing list and nearly a dozen people began participating in the co-development work. This initial shared development process was fruitful, with a number of patches and new approaches integrated into the MyLibrary model.

Conclusion

This brief history of the development of MyLibrary@NCState is just the first chapter in the continuing story of the "MyLibrary" model. As this volume illustrates, development continues with an even higher range of functionalities, greater reliability, and a ever-growing community of investigators and developers. It was not a fanciful dream in 1998 to assert the tremendous value of such a tool as MyLibrary to the NC State community, today it is an equally valid claim applicable to a worldwide community. In concluding one report on the NC State service in 2000, two members of the team noted that it was needful "to emphasize the fact that development and deployment of such a service is resource intensive --particularly vis-a-vis staff time." That is no less true today; however, the continuing development of the MyLibrary portal and its ever more refined feature set, as documented in this book, are the specific results of that inquiry begun in 1998.

Author's Note:

This essay builds on two previously published articles: "My Gateway: Personal Library Portals." by Keith Morgan in The Encyclopedia of Library and Information Science 2nd Edition, Marcel Dekker Inc. 2002; and "Pioneering Portals: MyLibrary@NCState" 198 by Keith Morgan and Tripp Reade in Information Technology and Libraries, 19:4 (Dec. 2000), pp.191-198.