April 2012

Implementing LectureBank: a Network to Connect Researchers and Scientific Event Organizers

Gregory Philip Dracoulis
Worcester Polytechnic Institute

MXC-0354

IMPLEMENTING LECTUREBANK: A NETWORK TO CONNECT RESEARCHERS AND SCIENTIFIC EVENT ORGANIZERS

A Major Qualifying Project
Submitted to the faculty of
WORCESTER POLYTECHNIC INSTITUTE
in partial fulfillment of the requirements
for the Degree of Bachelor of Science

by

Gregory Dracoulis

Date:
26 April 2012

Professor Michael Ciaraldi, Major Advisor

Professor Luis Vidali, Co-Advisor

Abstract

This report discusses the development process, design choices, and certain aspects of the implementation of key components behind LectureBank, a network connecting researchers and academic event organizers. We examine the tools involved and how the purpose and audience of the site as well as best practices and advice influenced the project. Additionally, we survey the latest advances in the field of web design, including new semantic HTML5 elements and microformats, and explore how we integrated and took advantage of them.

Acknowledgments

This endeavor could not have been accomplished without the splendid support of my advisor, Luis Vidali, who initiated the project and has stuck with and continued to believe in it. Invaluable assistance was provided by the National Science Foundation, who funded some of the work on the project over the summer of 2011, and Worcester Polytechnic Institute, who sponsored me to attend the SACNAS 2011 conference to publicize the service, make connections that will help us expand in the future, and start collecting feedback. I must also thank my major advisor, Michael Ciaraldi, who has been a wonderful asset and resource throughout this experience and who also believed in me enough to be willing to take me on as an advisee on very short notice. Finally, I would especially like to express my gratitude to my family and friends for their constant encouragement, motivation, and support, particularly my parents and my friends Amy Smekar and Lauren McClain who have been there for me throughout the whole process, offering vital understanding and assistance. I sincerely thank all of you for making this possible.

Gregory Philip Dracoulis,

Worcester, Massachusetts, April 2012.

Table of Contents

Introduction

This paper discusses the development process, design choices, and certain aspects of the implementation of key components behind LectureBank, a live web application which serves as a networking tool that connects scientific and academic researchers and related event organizers nearby geographically and with related interests or fields of study. The goal of the site is, as articulated on its home page, to "provide a networking tool for students, faculty, and researchers organizing lectures, seminars, conferences, or symposia to find qualified candidates to give original and interesting talks at their events" and "level the scientific playing field by creating more opportunities for postdoctoral researchers to bolster their credentials and improve their speaking skills," promoting general scientific literacy and wider interest in scientific careers in the process (LectureBank, 2011).

The site is intended to allow for the creation and discovery of opportunities to speak, and to facilitate the exploration of new talent for lectures, seminars, conferences, and symposia. It aims to enable event organizers to search for local researchers whose experience and knowledge will enhance the event they are planning, read brief abstracts summarizing the potential talks each candidate can give, and get in contact to invite people to speak live and in-person. On the other side of this, it allows researchers and potential speakers to post profiles detailing their papers and lab experience, interests, and past and future speaking engagements or presentations in order to increase their online visibility, since CVs posted on lab websites are often buried deep within the complicated hierarchy of a university website and not optimized for search. The site is also designed to help researchers find opportunities to speak, giving them a chance to gain experience and the confidence that comes with it. Finally, it allows the public to see the talks and events

happening in their area, to hear about the most cutting-edge research topics in science, technology, engineering, and math directly from the researchers themselves, and to meet other people with similar interests by attending conferences and lectures at local universities.

This document focuses on detailing what exactly makes LectureBank something new, unique, and special. There were many decisions that had to be made and considerations that had to be taken into account during the development process, including at the most basic, what tools to use, where to host the site, and how to deploy it. Further on, we examine some areas of the site—like the registration and search systems—that follow longstanding best practices, and we discuss how we can adhere to these ideals better today using more modern web development techniques and how those principles influenced our decision-making during the implementation process. We also take a look at some other areas, such as recommendations engine design and result ranking algorithms, that make use of concepts new enough that good resources explaining them are few and far between, and we show how this, too, influenced the final product. Finally, we survey some of the latest developments in the web design and programming sphere that allow us to help other services understand the structure, flow, meaning, and content of each page on our website better, and that allow us to open new interfaces to other developers and allow them to use our content and services in a federated, well-defined, programmatic fashion.

Background

Motivation

The initial idea for LectureBank came out of a green-field project[1] to find ways to support and promote underrepresented minority scientists. When looking into the literature related to this topic, we saw that even now, white males continue to be overrepresented across the spectrum in Science, Technology, Engineering, and Mathematics (STEM) fields—in undergraduate admissions, bachelor's degrees awarded, Ph.D.'s and full professorships held (Brommer & Eisen, 2006). This is problematic because these statistics are not keeping up with the changes in percentage of minority groups as part of the United States' population as it continues to climb. There are further worrisome implications as well; right now STEM fields in the US are facing a "brain drain" the likes of which we have not seen before (Nelson, 2009). Students are abandoning STEM programs at high rates just as many of the baby boomers that comprise much of the STEM workforce are about to retire. In our previous report, we concluded that "in neglecting our own US-born minority citizens, we are neglecting a huge pool of talent that could be effectively mobilized to help close the gaps in our graying STEM workforce" (Dracoulis, 2011). This reaffirms the need for systemic change on this front to preserve the socioeconomic progress that excellence in STEM fields has brought us over the past several decades.

In observing and analyzing this problem, we found there are many causes for underrepresentation, including the immense financial burden of higher education coupled sometimes with the desire to seek fields with a better return-on-investment. There is also a lack of a coherent support apparatus, which lessens the perception of opportunity for

[1] A software project in which the best solution is not known or suggested, and there are no constraints imposed by prior work.

minority groups in STEM. Language barriers for those with a first language other than English, and the perceived dissatisfaction of mentors or role models with similar backgrounds or, in some cases, a glaring lack of relatable advisers have all further influenced the disparity (Dracoulis, 2011). There are many disaggregated programs designed to combat parts of this situation, but most of them are unilateral, institutional efforts that are too isolated to have much of a systemic effect. We eventually decided to look just at the problem of reducing attrition at higher levels of the "academic pipeline" (namely the postdoctoral level) as underrepresented minority scientists are starting to transition to the next phase of their careers. We expected that by increasing entropy within the scientific community at upper levels by creating a tool that would connect postdoctoral researchers to event organizers looking for speakers or presenters, we could increase the number of visible opportunities available and thus improve morale. We also sought to increase the level of "connectedness" in the STEM community by designing the tool to put people together in person and add to the roster of researcher's contacts at institutions other than their own, and provide good practice for speakers to hone their skills and become more confident and comfortable presenting. Our hope is, if we can accomplish this, that it will propagate an improved outlook throughout the STEM pipeline.

Our search tool would, in addition to being useful in and of itself, create a more level and diversified playing field by forcing those involved to "search based on qualification instead of relying on their comparatively homogenous real-life social network" (Dracoulis, 2011). According to a report by Bramoullé and Rogers, et al. (2012), searches based in an online networking tool such as the one we hoped to build tend to reduce homophily, or "love of the same," our often subconscious tendency to gravitate towards linking with

people similar to ourselves. The paper explains, "this mechanism is not obvious a priori, given the presumption that social segregation may be aggravated by network effects. Yet it has an intuitive explanation. Even if meetings across groups are relatively rare, a new individual still meets some individuals from the other group. Through them, he then has access to many additional individuals from the other group, and so diversity among friends of friends is greater than among friends. We believe that this captures an important mechanism at work in real networks […]" (Bramoullé, Currarini, Jackson, Pin, & Rogers, forthcoming 2012). In this way, we justified the creation of a networking tool like ours, and we hoped to exploit this phenomenon behind the scenes to achieve our goals.

Overview of the Tools

Development

A number of tools were instrumental in the implementation of this site. In particular, we used PHP and MySQL and developed on a Macintosh using Adobe Dreamweaver CS4. In this environment, it was easy to set up a testing server on our local machine to try out versions of the site without having to upload the changes and test them live. We used the free, popular MAMP (for Mac, Apache, MySQL, PHP) package available at http://www.mamp.info to run that server. Even though we only had one primary developer working on the source code, we decided to make use of a version control system for source control, backup, annotations, and deployment so we would be able to roll back changes on different servers quickly if any errors occurred as a result of a code update. A comparison of using Subversion (the version control system we used) and FTP upload for deployment can be found in Appendix A, and instructions on how to actually use our deployment system can be found in Appendix B. The Subversion server we use is provided by Worcester Polytechnic Institute for institutional use at http://sourceforge.wpi.edu.

We created a second “staging” subdomain on the same server as the live site and password protected it so it would only be accessible to authorized users; we use this to test any changes a second time before pushing them live, since there can be some differences in configuration between our local testing server and the production system. This has saved us some headache over the course of the project, by allowing us to identify problems and test fixes in a safe, private environment; for instance, on one occasion we noticed on our live host we did not have the ability to create views in our database and so had to reformat a number of queries we had created and modified, and in another incident we realized the live host had certain PHP functions we were using disabled by default for security reasons, so we had to override those settings before making our changes, giving us time to restructure our code to use different, more secure built-in functions instead. We have one test database that resides on our local machine, updated by dumping data from the live database every so often, and we use the free Sequel Pro tool available for Mac at http://www.sequelpro.com to manage it. We also have two databases on the live server- the production database, of course, and a test database used by the staging server and updated in a similar manner to how we update the local database; we often use PHPMySQL to interact with these. In addition to these, we have a fourth, backup database that resides on WPI’s servers and is updated every five minutes by the following cron script that runs every five minutes on the main server (note that there are no spaces between **`-p`** switches and passwords):

```
mysqldump --add-drop-table -h {hostname of primary DB
server to be backed up} -u {username for primary DB
server} -p{password for primary DB server} {primary DB
name} | mysql -h {hostname of backup DB server} -u
{username for backup DB server} -p{password for backup
DB server} {backup DB name}
```

Hosting & Acceleration

We host the site on a "cloud" host, MediaTemple. Using their (gs) Grid Service spreads our site load across multiple servers for both the Web and the database component, with each bursting as necessary based on traffic. This frees us from worrying about scalability and larger capital investments in infrastructure; a certain amount of bandwidth, storage, and CPU time-sufficient for 90%+ of websites-is made available, with metered billing for usage beyond those limits. Although it is a multi-tenant environment, we are also provisioned with shell access to our own segment of it, which gives us greater control over configurations than a typical shared host. Thus, we can deploy a sophisticated service with minimal cost, and we can be confident our hosting package will grow with us.

We also make use of a Content Delivery Network, or CDN. This replicates components of our site across servers worldwide, minimizing latency, load times globally and reducing main server load by aggressively caching static resources. It also speeds up the site by compressing images and "minifying" (i.e. refactoring and text compressing) JavaScript and Cascading Style Sheet files. Because it takes snapshots of requests and resources, the CDN can also provide offline access to certain pages even if the main server goes down. The one we use, CloudFlare, is free and additionally provides a suite of security

features like automatic email address obfuscation (where the email address is visible to users but scrambled to spam bots or crawlers) and presenting CAPTCHA challenges on detection of suspicious behavior or an IP address flagged elsewhere as suspicious. The fact that the service integrates seamlessly with MediaTemple, our host, increased the appeal of using it even further.

Methodology

Registration

Keep it simple, don't hide it

Registration is a key component of any site that wishes to have users logging in to and using it. Simplicity and visibility, across a whole service's design but in this engagement particularly, are paramount. We want to prompt users to register with us, and for this we need to make how to create an account as obvious as possible—we do this in LectureBank by showing a large green "Sign Up" button on the home page. Of course, even if the visitor ignores that prompt and continues browsing the site, or if he or she landed on a different page within the site's hierarchy, we also want the option to sign up to be readily available and apparent. We do this by keeping a link to sign up accessible from each page on a panel that detects whether or not a user is logged in; if so, a so-called "reflector card" is presented, letting the user know which identity is logged in and giving them access to their profile and other tools, and if not, a login form and a link to registration are made obvious. Although we have designed the site to keep as many aspects accessible without registration if possible, we visibly prompt for login or signup when a user tries to do something that requires it. We also have made an effort keep the user in the same context throughout that process, unobtrusively returning them to the page they were trying to visit or action they were trying to complete after they successfully login.

Provide a choice in how a user registers

It is a recommended practice now to--at the very least--explore the suite of single-sign-on (SSO) services available that allow developers to hook into a user's existing social identity on another platform or application. Facebook Connect is easily the most popular of these options, so we explored it first. However, based on responses to surveys and in

interviews we conducted, we felt it was a bad fit for this particular network. Many people we talked to, especially in the scientific and technical community, want to keep a clear demarcation between their work life and their social life, and they seem unsure of their ability to trust Facebook to keep that boundary as solid as they might like or need

Google+ also seemed like it could be a good fit, but at the time of this writing there was no readily available API for integrating someone's Google+ identity in the same way as Facebook. This is something that should be explored in the future, as further iterations of this API are released. There are benefits to associating one's Google identity with his or her profile on external sites. Google has already created tools to integrate and link authorship information that should be of some use in creating richer and more useful search results for our site across the web. Google+, with its Circles feature, keeps much better defined boundaries between social groups. It seems as if this feature would make using Google+ more palatable to our audience than using Facebook Connect, were we to provide the option. Our current problem is that the state and trajectory of Google+ is unclear. Additionally, with Google's reputation for abandoning products if they do not get popular enough (see Buzz), we were discouraged from spending energy and resources integrating with Google+ when there is a chance that the application could get dumped. In light of these problems we have considered simply connecting to Google accounts (which provides federated logon but little in the way of identity information), while waiting for Google+ to become more firmly established.

LinkedIn or Indeed registration might also be something to explore, since these communities are built around creating professional identities and seeking opportunity. Other identity providers include Twitter or Windows Live, but these are less popular than

LinkedIn and Indeed in general, and seem to be lacking even further in popularity with our target audience. The question here is, while breadth of choice is important, how much time and effort should we be investing on that breadth when the returns may be minimal? Our registration and identity provider choices should reflect the tone of the network itself.

User Interface Counts

In order to simplify the registration process, we have a one-page signup form that strives to keep information streamlined and joining the site as convenient as possible. We carry out a wide range of validations while the user is typing—this way he or she does not even have to click a button to determine whether the form will go through. We check the username for length and uniqueness every time the user presses a key in that field and as soon as they're done entering a name, they know if it's available or not; they can conveniently correct errors without filling out the rest of the form, having to wait for it to load and process on the server side, and finally being returned an error message and having to retype both passwords. This keeps the user engaged and reduces the time they have to spend registering, as well as frustrations with our site.

We perform similarly advanced validations on the email field as well; on every keypress while the field is in focus, we call a function that first checks for proper length, then checks the string inside the field against a regular expression (the expression `^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*(\.[a-z]{2,4})$` to be specific) to ensure it is of the right format. Third, we take the part of the supplied address after the "at" symbol, "@,"and query the Domain Name System to check for the existence of mail servers at that address. This allows us to catch cases where an address might seem to be supplied in the proper format, but is not actually registered or functional

(e.g. “test@nonexistent.co.jp” where “nonexistent.co.jp” would be a valid URL, but has no MX records). Finally, we check the address for uniqueness in our database before displaying a live, green checkmark indicating to the user that their input has been accepted. In order to assist the user in creating a secure password that meets our requirements, we offer a final set of validations that first checks the length of the password and then, once it is of the necessary length, offers a strength rating (weak password, indicated in amber; good password, indicated in light green; or strong password, indicated in dark green) using the algorithm specified here: http://phiras.wordpress.com/2007/04/08/password-strength-meter-a-jquery-plugin/. In using this system, we gently nudge the user in the right direction with regard to their password choice whilst not being too restrictive. Finally, we verify that both the password and password confirmation fields match before form submission so the user can correct any discrepancies.

Once the form is submitted, we carry out all of these validations once again, but completely on the server side. It is always prudent to sanitize and double check input from a user instead of blindly trusting client-side validations and inserting potentially unsafe or incomplete user-controlled strings into the database. This also sidesteps any problems that may arise from JavaScript not being enabled on, for example, a particular mobile device or outdated browser. Even with client-side validations, the form does nothing to stop users from submitting incomplete data; it merely advises them in advance that it will not be allowed to pass through the gates on the server-side and will be returned to them. If there are errors, the form page will simply be reloaded with the offending fields prepopulated (passwords excepted), highlighted in red, and error messages displayed in the same locations above those fields as they are on the client-side. The client-side validation scripts

were also written in such a way that they interact properly with any server-side error messages displayed, updating them inline and live as necessary when the problems are corrected.

When the form is finally submitted correctly, we start the process of verifying the user's email address. We do not allow logins or further registrations using the email address until this process is complete. We chose to use this system instead of a CAPTCHA, even though it breaks the flow by requiring the user to briefly exit our application, because it is crucial in the context of this network that we have correct and valid email addresses for our users, who use email as one of their main modes of communication with others in academia. We carry this verification out by adding a tuple (row) to the users table with the field "status" set to the value "verify," which is checked for at login and whenever user information is queried (those with "status" set to "verify" are excluded from the results sets), and we create a unique activation token by taking the SHA1 hash of concatenated random strings. We construct an email including a link to the activation page with the token as the query string, and fire this off to the user. When the user clicks the link, they are taken to the activation page, which checks the supplied query string against any active activation tokens in the database for users with "status" still set to "verify." If a match is found, the system sets the "activationkey" column for that tuple to NULL so the token cannot be reused or recovered, and the "status" column is set to "activated," enabling the user to login and use the site. This describes the flow and interface regarding registration with LectureBank.

Watch, Learn, and Adjust

Analytics can be important and useful to measure and hint at adjustments to virtually any aspect of a website. However, it could be argued that they have the power to make the most difference at the registration phase. By tracking conversion and drop off rates, as well as user preferences among identity providers, websites can make adjustments to how the registration page is presented (Janrain Inc.). For example, if, over time, a significant percentage of site visitors register with a few specific providers, website administrators can choose to highlight those on the homepage. Signup metrics can be tracked with respect to different positioning, placement, sizing, and even color of calls-to-action, different signup forms, or even simple adjustments to front-page copy. Administrators can look at which pages people land on most often, what external site or location they are coming from, and which paths of action result in more (or fewer) signups. They can even try to identify what exact points of friction lead to a particular pattern of aborted registrations. For example, in LectureBank we are able to track which accounts were abandoned at the email address verification stage and for how long; this gave us the opportunity to send follow-up emails after several days to try to re-engage the user, or to check our mail systems to make sure an activation email was actually sent. This mechanism actually enabled us to troubleshoot our systems when we noticed an anomalous set of users never activated, where they normally do so quickly after their registration form is submitted. We thought to run some checks on our email system to make sure our messages were not being flagged as spam, however as it turns out, we had changed the password for the email account we were using and had forgotten to update it in the code. We immediately corrected the problem, and manually generated new activation codes and

messages for each of the users, bringing several of them back into the network where their registration might otherwise have been left to die. We also use Google Analytics to detect such parameters as language and browser preference, social engagement, search terms, and landing pages so we can target our attempts to highlight particular terms and optimize for particular browsers over others, as well as see which pages we might place a more prominent call-to-action for registration. Drilling down into network (Internet Service Provider) information gives us relevant information as well, as it can show us from which institutions (i.e. colleges and universities, or research labs) users are accessing our site, allowing us to target particular organizations or campuses more effectively. We can also create “funnels” that show us the paths that users take through our site to different “goal” levels, which can show us information such as how people get through the site to the signup page, how many make it from the signup page to successful registration, and how many make it from a successful registration to clicking the activation link in their email and login for the first time. Social referrals and conversions are both also trackable, which are useful in deciding which networks you choose to integrate more tightly with.

Making the Most of User Data

One of the most important considerations in a registration system is just what information to collect and why you are collecting it. A number of sites tend to overcollect information, hoping they might be able to use some of it in the future. We decided this was the wrong approach to take with LectureBank, and instead opted to collect a minimum of data, only offering prompts for information we would be positively justified in retaining. This minimizes the information we have to store and protect, and we consider this a best practice for our site. We strove to make as many fields across the site optional or "opt-in"

as possible, reducing the time it takes to register and fill out forms. Instead of forcing users to fill in their profile information as they register, we allow them to skip that step, instead opting to encourage the completion of their profile at their leisure by displaying messages reminding them their profiles are incomplete and showing links to the forms that allow them to fill out their profile after they log in. This means making the site more flexible, as we cannot guarantee what information may be present or absent on a given profile because little information is required to create an account or another entry (we may not even have a user's real name). However, we believe the extra effort in striving for this flexibility is worth it to increase convenience for our users and allow them to make the decision about what information to entrust to our site, instead of trying to push that on to them. The primary reason we collect the data at registration is made clear on the registration form: the username serves as the new user's address on the site; we use their email address for verification and site related communications, such as "forgot password" emails, as well as for display to other users so they may be contacted; and of course the password protects their account. By streamlining registration, we hope to increase the number of users that successfully complete the registration process by making sure we do not test their patience. Other information not necessary for establishing an account is only collected later, after their first login, as part of a different form to customize the user's profile and other elements on the site. This step can also be skipped if the user wants to dive straight into using other elements of the site. For example, a user looking to simply recruit someone for a talk or conference via LectureBank might not even want to create a profile, but instead just take advantage of some of the filters that can be created when logged in (e.g. location filtering based on profile ZIP code). While we do not require any of this information to be

entered, when a logged-in user encounters a section that could be enhanced with relevant data, they are reminded that it is missing and given a link to enter it directly. We hope this increases engagement on the site and think it is a better decision to give users choices with regard to how much information they enter into their profile (albeit along with justifiable reasons and prompts informing them why we think they should); forcing users to enter information before they realize why it might be useful can result in malformed, polluted, outdated, falsified, or incorrect data, which we find is often worse for the management and presentation of the site than no data at all. Some of these challenges can be overcome by providing autocomplete mechanisms where possible, as we do for tag and institution entry, to speed up data entry, minimize misspellings and junk data, and suggest ideas and the proper format while remaining flexible. Others can be met by tying in to existing profiles which are often kept up to date and have a greater chance of containing correct information, but as we explored earlier, this method has its own set of considerations and limitations. Often these profiles are complete to varying levels as well, so it is of no harm to have the flexibility to deal with such cases already built in. In the future, it is likely we will explore the option of integrating with one or more networks or social data sources to pre-populate forms or make suggestions to further increase engagement and expedite the process of creating a profile, as we continue our practice of letting the user decide what to actually submit to our database.

Sorting & Suggesting

Personalized Results

One of the main thrusts of LectureBank is connecting event planners with potential speakers. If we are to achieve this goal, it is imperative we provide ways for these two types of users to find and connect to one another. This introduces the problem of

presenting relevant results, but we have to determine what exactly this means. How does one assess relevance? How do we know the assumptions we make will be correct? Only after we determine how to sort out and retrieve a set of results can we begin to address the problem of how to display them. Many services on the web have had to tackle this very issue in order to differentiate themselves—Amazon, Google, and Netflix are prime examples. It could be argued that their business depends on their sorting algorithms. Amazon's recommendations engine is a huge driver in its success; Amazon claims approximately thirty-five percent of its sales are generated by recommendations, making that system second only to direct search in the hierarchy of technical contributions to the bottom line (Marshall, 2006). Google's very existence is predicated on the promise that it can guess what you're looking for better than any other search engine. Netflix's recommendations system has been such a competitive advantage that the company famously offered a one million-dollar "Netflix Prize" to the first team who could improve upon the accuracy of their Cinematch algorithm by at least ten percent (Netflix, Inc., 2006). So, given the important role of good recommendations in creating a compelling dynamic site today, we set about trying to determine where and how to start building a system that would deliver them.

We quickly determined the existing literature on the subject is sparse and limited in its usefulness. Overviews of data mining techniques and the suggestion engines available to us trended toward the extremes of simplicity and excessive academic depth. In retrospect, this is unsurprising as the particulars of many algorithms, being so critical to a business' competitive advantage in the online sphere, are closely guarded trade secrets and the general concepts underpinning them make up a developing (and as yet disaggregated) field

of study and involve a good deal of advanced mathematics. In short, it is very easy to go extremely deep in this area but this was outside the scope of this particular project—it could easily be the subject a degree project all its own. In this light, we decided to eschew complex statistical techniques in favor of starting with more trivial suggestion methods. This leaves much to be desired, but current algorithms do not provide large enough improvements over intuitive best guesses using simple linear modeling or averages to justify such a substantial initial time investment on our part given our limited resources. Netflix's original Cinematch algorithm, as an illustrative example, only provided a 9.6 percent improvement on root mean square error (RMSE) over using just averages across the whole Netflix Prize test database as naïve predictions for a user's rating of any particular movie (Netflix, Inc., 2006). This, in our view, provides actual statistical justification to our more simplistic approach when both marginal cost and marginal benefit are considered.

Serendipity and Fairness

A major consideration in how we engineered our suggestion systems was the *purpose* of LectureBank, which on a high level is to increase entropy and connectedness within the scientific and academic community. We want to introduce people who may have tangential connections to this community, encouraging the cross-pollination of ideas both within and across disciplines. Additionally, it is one of our stated aims to reduce homophily—a preference for similarity, conscious or unconscious—within the STEM community; thus we are careful not to overfit or overpersonalize. Our views are summed up well by the words of John Stuart Mill:

> "It is hardly possible to overrate the value… of placing human beings in contact with persons dissimilar to themselves, and with modes of thought and action unlike those with which they are familiar… Such communication has always been, and is peculiarly in the present age, one of the primary sources of progress" (Mill, 1909).

While the site mainly serves as a facilitator for scientific and academic events, this function serves as a means to a greater societal end. Put simply, our overarching goals in creating LectureBank are to level the intellectual and academic playing field to minorities and create more "collisions" of knowledge, expertise, and opportunity. These desired implications were at the forefront of our design and implementation process. We not only had to preserve fairness and serendipity, we engineered it. As Eli Pariser says in *The Filter Bubble,* "In some cases, letting algorithms make decisions about what we see and what opportunities we're offered gives us fairer results. A computer can be made blind to race and gender in ways that humans usually can't." This is exactly what we want to accomplish! But, Pariser continues, fairer results are accomplished "only if the relevant algorithms are designed with care and acuteness. Otherwise, they're likely to simply reflect the social mores of the cultures they're processing—a regression to the social norm" (Pariser, 2011). Sometimes, depending on how they are programmed, computer systems can be even more discriminatory than humans since, while they are not forming explicit identities, the algorithms behind them must create implicit classes to make any sort of categorization. If those behind the system do not look beyond the numbers and into the potentially problematic groupings that may emerge, they risk, according to NYU sociologist Dalton

Conley, "reducing the complexity of near-infinite choice" and creating categorizations that "are more insidious than the hackneyed groupings based on race, class, gender, religion, or other demographic characteristic" (Conley, 2008). The problem here is that users don't even know what bin they have been thrown into, and so they can't make any attempt at challenging or correcting the machine's assumptions, and as Pariser points out, "because personalized filters usually have no Zoom Out function, it's easy to lose your bearings, to believe the world is a narrow island when in fact it's an immense, varied continent" (Pariser, 2011).

Two Primary Audiences

With all the above in the front of our minds, we set about determining exactly what needed to be sorted and why. The first step in this, of course, is considering your audience. Our concrete objective to facilitate connections between event planners and potential speakers, when expressed as such, does a good job of making a clear distinction between two disparate use cases: planners looking for speakers, and speakers looking for opportunities. While this is a very useful distinction, we must keep in mind it is a fluid one, not an absolute. As such, we designed the site to be flexible: if you create an account, you can skip populating your profile and go directly to posting an opportunity, and you can tab back and forth between roles at will. We had to make a conscious effort to ensure users would not be locked or boxed into one role, acknowledging that they might want to switch depending on their career stage or just their needs or inclinations at any given point in time. In fact, we endeavored to make the site useful even in its default "neutral" mode, where any kind of user can establish an online academic portfolio.

Having established the major motivations for using the site, we moved on to defining important usage scenarios. These included adding information to one's portfolio (perhaps in hopes of being discovered by a search) and actively looking through a list of opportunities on the speaker side, and actively searching for a speaker for an event or passively posting an opportunity or call for submissions on the event planner side. By running through these scenarios (included as appendices), we were able to establish what to suggest as matches to each group: potential speakers would likely be looking for opportunities, and event planners would be looking for speakers. With this established, we could look at each of the entities—opportunities and speakers —and seek out relevant information that we could use to present results. The dimensions that were eventually decided upon naturally separated themselves into another pair of groups —those that required information about the logged in user and those that did not. This was purposeful, as we wanted the site to remain usable if someone is not logged in and casually browsing. In this case, an unregistered user would be able to see the network's promise, but the design still encourages sign-ups by holding some of the more interesting tools only for registered users. We decided all visitors using the "For Speakers" tab of the site would be able to see opportunities that were recently posted or that just opened for submissions (i.e. became active on the site), and all visitors using the "For Planners" tab would be able to see the most active presenters (those that had the largest amount of populated profile information pertaining to speaking qualifications) and the most recently active presenters (people who recently updated their basic profile information or posted speaking engagements or research to their profile). Logged-in users would, of course, benefit from carefully considered filters on top of those just mentioned; those acting as Speakers would,

if they provided geographic information (like a ZIP code or home institution), be able to see a list of opportunities sorted by the distance from their home location to the event venue, and would also be able to see opportunities with tokens "like them." This is the most intricate bit of our suggestions system: we take as much of the user-supplied information as we can and split it into tokens or one-dimensional "genes" by allowing user-supplied tags for their general professional interests, as well as their talks and publications to act as items to match on. We also take the titles and abstracts of each of their speaking engagements or research items and remove stop words to populate the search set and develop an idea of their background (Iskold, 2007). We use basic self-organizing classification now, but once more populated this dataset can be mined to find and predict more implicit relationships and similarities based on either a learned or specified taxonomy. We use the data the logged-in user has provided to execute a search on many dimensions—if a Physics researcher even cursorily mentioned using a genetic algorithm in a talk abstract, an opportunity at a plant genome conference might show up in the result set —and we then randomize the order in which the results are displayed each time to preserve both interest and the element of serendipity in the design. We match the same way when a logged-in user acting as a Planner has posted an opportunity looking for matches—any tangential association is explored by the system and might result in a hit, and the result set is again randomized. We also present these results as only a section of a page, along with less "intelligent" suggestions in order to try presenting something new and potentially interesting each time.

Searching

Facets

The other natural activity a user may wish to engage in, besides browsing, is directed search. This system must be designed with the same level of care as the browsing element. As Jakob Nielsen states, "search is the user's lifeline for mastering complex websites" (Nielsen, 2001). Not every website needs search, but one such as ours—a large network with many data items that will likely grow in the future —can benefit greatly from having that functionality available up front (Fekete, 2008). But what should the scope of the search be? We have to establish which items of data we will be searching before we proceed any further. It can be easy to get carried away with designing advanced search features, result display algorithms, and the like before considering something that is admittedly so basic. But content within any complex site can be categorized by several independent facets; for example, in our network, we have a number of different entity types that are important: people, places (institutions), events, research items, and opportunities make up the list of critical content pieces across LectureBank at the moment, though such a list is always subject to growth over time (Taxonomy Strategies, 2004). It was necessary to provide different search harnesses for all these items. However, this does not mean we had to—or that it would be best practice to expose individual search facilities for each. According to Nielsen, we must approach this question of search scope with caution:

> "Scoped search lets users limit the search to results from specific areas of the site (the search scope). In general *this is dangerous.* Users often overlook the scope, or they think they are in a different site area than the one they are actually searching (Nielsen, 2001)."

As websites continue to trend towards greater and greater complexity, and especially when multiple services or use patterns become available in a single site, scoped search can be sufficiently useful—if implemented in a considered fashion. Areas of the site offering scoped search ought to be clearly delineated and have a specific problem to address, and the default search scope should be kept to the entire site (Nielsen, 2001). As such, we designed our main search tool to cover all of the important facets available (as Apple and Facebook are doing with their single, comprehensive search boxes that span discussions, songs, apps, news, and more) and decided to separate out the results by type. As we are able to tokenize most of the data in our database as well, we chose to optimize for the "preponderance of simple searches" and emphasize our ability to handle single-word and very short queries while still producing quality results (Nielsen, 2001). Further, since we have already established the existence of disparate and unique audiences and use cases for our site, we did opt to develop scoped searches targeted at planners and speakers, which emphasize people and opportunities respectively.

Mixing Search and Navigation

Although search is often quick and to-the-point, this is not always the case. Sometimes users do not know exactly what they are looking for, and others are using search as a method of site exploration. In any case, it is our job to endeavor to make the

search system as useful as possible, and while things like meaningful titles and descriptions are obviously important, “one of the keys to improving discovery on a site is allowing a mixture of search and navigation” (Taxonomy Strategies, 2004). So, although the results themselves are of course important, they are not necessarily enough in and of themselves. We should facilitate refinement of a search by providing related terms. It would also be advisable to go beyond simple titles and descriptions when returning a result set—we should provide inline links to relevant information that might otherwise be buried deeper in the site, links to related content within our subject taxonomies, and allow the user the freedom and opportunity to explore a bit more fluidly (Taxonomy Strategies, 2004).

With these recommendations and ideas clearly articulated, we set out to realize them. Results in each category or “facet” are augmented with expandable sections offering a greater look into the content that lies within them, exposing more key information at a glance while also providing links to further explore them in turn. Each tag that is listed anywhere on the site, but most notably on the search result pages, links to a search query for matching results; listing these along with results provides an easy way to create search suggestions without having to implement any sort of fancy system to determine relatedness—the results themselves may be tagged with synonyms or more indirectly related terms and form a taxonomic network of their own, which the user can then explore with a click. Finally, we provide some additional site navigation in results as well, allowing users to discover and explore people and other entities even if they are browsing more along a different dimension. For example, when authors are listed, we link to their profile pages and when institutions come up we list some of the people associated with them. These richer results are an attempt at driving engagement and reducing abandoned or

unsuccessful searches by giving users ideas of how to hone their queries without having to reformulate them and providing them with more options to consider.

Structure and Meaning

HTML5

Although you would not necessarily know it if you looked at a modern website today, initially HyperText Markup Language (HTML) was designed to mark up text documents with some meaning. Over the years, web developers started making use of certain HTML elements not for the meaning they conveyed, but instead for how browsers presented those elements. This became the status quo for a time, but eventually objections to this widespread practice were raised and the standards bodies involved reacted by creating new, essentially "meaningless" elements like **`<span>`** and **`<div>`** to give developers the resources they needed to create logical groupings without making erroneous implications (mostly for presentation control). This too, for a while, was fine. However, as computers have become more powerful, and more and more information has shifted onto the Web, we are finally at the crossroads where we really need to sift through and interpret the content of this ocean of data in more depth and we actually have the horsepower to do so on the requisite scale. Enter the newest iteration of the HTML specification, HTML5, and its "semantic" HTML elements (Matthijs, 2008).

These new elements are being introduced alongside the old standbys—augmenting rather than replacing them—so they remain available for use when indicating any particular meaning might be unwise or unnecessary. However, they do allow developers to have much more to work with in pointing out what parts of a document contain certain kinds of information. This enables us to give some sort of "tone" and "shape" to a

document—to indicate which sections contain the most important content, and those that contain control elements that might be better deemphasized or ignored in certain contexts (de Valk, 2007). Screen readers, search engines, and programs like Readability and Safari Reader can use the new information at their disposal to better interpret the structure of a web page, instead of using complicated heuristics to guess. Segments marked up with the new **<section>** and **<article>** tags can be assigned greater weight and extracted to create better search result snippets, **<hgroup>** tags can be used to keep parsers on track when they are trying to build a document outline, and **<header>**, **<footer>**, and **<nav>** elements can be automatically skipped or removed, making clunky "skip navigation" techniques formerly necessary to improve accessibility obsolete (Pilgrim, What Does It All Mean?, 2010). Good semantic HTML provides the guidance necessary for computers to determine a page's structure and topic, and because of this search engine spiders do appreciate it—they can better understand what a page is about and make more accurate categorizations based on that information (de Valk, 2007). Because of these implications, we thought it worthwhile to put work into using these new semantic elements whenever appropriate across all of LectureBank.

Microdata

Although it can also be considered semantic HTML, microformats and microdata serve a different purpose. Where semantic HTML5 tags in general broadly describe the content of a section in a document, microdata can be used in tandem to give the content a more specific context and make portions of it machine-readable. For example, you can use semantic tags to describe a particular piece of content as an article, while you can use a microformat vocabulary to indicate that it is a scholarly article enclosed within that tag,

and further to use an agreed-upon format to make it easy for a computer to extract the author, journal, and publication date. The key element of a microformat is its shared markup vocabulary—the fact that multiple services can expect and understand certain organizational hierarchies and individual tag elements (Pilgrim, "Distributed," "Extensibility," & Other Fancy Words, 2010). Although there are several competing standards, the emerging victor is the relatively new schema.org microdata initiative. Google, Bing, and Yahoo all collaborated on developing a common structure and vocabulary, and Google in particular has selected it as the preferred microformat and uses information gleaned from this markup to improve search result snippets even further by pulling out data including ratings information, names and authors of articles, and the time, date, and location of events, and presenting them in clearer and more useful ways (Guha, 2011). As with the new elements introduced with HTML5, we have made every effort to provide richer, machine-readable data using these schemas where possible.

Link Relations

One final change introduced along with the HTML5 specification is the addition of new link relations. These tags allow webmasters to indicate the content of a page being linked to, and specify its relationship to the current page. Search engines and other programs can use this information to assemble a better model of how pages are coming together. For example, Google introduced the **`rel="nofollow"`** attribute so that user-provided URLs could be flagged as untrustworthy and search spiders could ignore those links in calculating rankings in order to try to reduce comment spam (Cutts & Shellen, 2005). The HTML5 specification also standardizes a complementary **`rel="external"`** to point out a link that leads outside of the current site (Pilgrim, The Road to HTML 5: Link

Relations, 2009). Google introduced the **`rel="author"`** attribute recently as well, in a bid to try to embed better authorship information in search results by resolving either to an email, a Google+ profile, or a user home page within a site (Hansson, 2011). Finally, **`rel="tag"`** is a relation that marks tags describing the current page; it initially gained popularity when introduced by the blog search engine Technorati, but just recently was also incorporated into the HTML5 standard (Çelik & Marks, 2005). We make use of all of these attributes, among others, as designed wherever possible in our site as well in order to provide the richest, most open experience to our users and other services across the web.

Exposing Services

OpenSearch

In that same vein, we hope to open other aspects of our site for integration and public consumption across the web. Sharing information aligns with our values and presents an opportunity to gain traction and recognition we might not otherwise enjoy. One of the technologies we use is OpenSearch, a protocol that describes a search engine so it can be used by other client applications. We present HTTP headers advertising the availability of the service, along with an XML description document that details the syntax for queries to our search engine and the elements that are to be expected in a response. When queried in the specified manner by an external service, we return a specially formatted response to the requesting application, which it can then parse. The resulting information can be used by the program to augment its own engine or the results can be looped through like a response from a simple API to find a particular item on the site (Clinton, 2005).

Really Simple Syndication

OpenSearch responses can be returned in the RSS or Atom formats, and so engineering LectureBank to comply with the protocol also had the advantage of presenting another useful front-end to our system. We chose RSS over Atom for its greater maturity and more widespread adoption and recognition. In line with the OpenSearch protocol, we created **`rel="alternate"`** links in the HEAD of search results pointing to a corresponding, specially augmented RSS feed. This enables autodiscovery of the feed, and many modern browsers (and RSS readers as well) will recognize its presence and point out the ability to subscribe to it. This ends up creating a cheap, easy way for users to set alerts or search agents on a particular query—the feed reader will check the results every so often and display a notification when it is updated and a new result appears. The feed can also be used in other ways; it could, for example, show the results of a relevant search term embedded on a page in a feed reader widget, among many other potential applications.

Future Work

Although a great deal of work has gone into the site and this project, there is still much more we can do to take it even further. As mentioned, we currently use relatively simplistic techniques to rank and rate both suggestions during browsing and results during search. We would like to be able to implement some more advanced ranking algorithms and use some data mining techniques to incorporate interaction data and perhaps user ratings into those calculations, and enable us to better predict what items on our site a user might find relevant, interesting, or worth exploring. Something that might help with this would be incorporating information available on PubMed like full text, citation networks, and information about collaborators to give us access to a larger dataset to work with as well as to make creating a profile easier since we could try automatically populating some elements of it. One other avenue that might be explored is integration with Wolfram Alpha and its API to potentially try to bootstrap a taxonomic network and make search suggestions (like "did you mean" and "related terms" augmentations). As mentioned early in this report, it may also be good to explore the idea of integrating more deeply with social networks. We currently provide Google "+1" buttons for different items across the site, but it would be good to be able to go further and actually integrate with larger identity providers like Google, Twitter, and LinkedIn so users do not have to create an account just for our site, and so we can make use of some of the sharing and other features of those networks APIs as well. On the other side of that, considering incorporating an OpenID service into LectureBank could also be a good idea, as it would enable single-sign-on into other services like third party comment systems so users wouldn't have to provide credentials again or create other unnecessary accounts.

The user interface could certainly benefit from some more work; the profile and events systems, in particular, are not quite designed to the same level as the rest of the site and could use some attention. Further, while we do provide a feedback mechanism already, it would be useful to conduct some empirical A/B tests to compare designs, and perhaps engage in some more rigorous user testing (e.g. webcam and screen capture studies) to identify any issues visitors might be having or friction points that could use improvement. Additionally, time could be spent developing a more mobile-optimized presentation for the information contained in the network, especially given the explosion in popularity of smartphones and other Internet-connected mobile devices. The event system could benefit from integration with a service like Eventful or Meetup to publish lecture information and handle RSVPs and possibly even ticketing if necessary, and it would be useful to integrate with YouTube and a document host like Scribd to enable users to embed videos of their lectures, full text searchable papers, or presentation materials on their profiles. The site could certainly use a robust commenting system, and it could be beneficial to incorporate a Pingback or Trackback system to find and track inbound links to a user's information and comments made about items in our database offsite. Finally, a security audit is due and highly recommended, just for the sake of prevention and safety.

Bibliography

Çelik, T., & Marks, K. (2005, January 10). *rel="tag"*. Retrieved February 26, 2012, from Microformats Wiki: http://microformats.org/wiki/Rel-Tag

Bramoullé, Y., Currarini, S., Jackson, M. O., Pin, P., & Rogers, B. (forthcoming 2012). Homophily and Long-Run Integration in Social Networks. *Journal of Economic Theory* .

Brommer, C., & Eisen, A. (2006). FIRST: A Model for Increasing Quality Minority Participation in the Sciences from the Undergraduate to the Professoriate Level. *Journal of Women and Minorities in Science and Engineering , 12*, 35-46.

Clinton, D. (2005, December 6). *Specifications/OpenSearch/1.1/Draft 5*. Retrieved January 29, 2012, from OpenSearch.org: http://www.opensearch.org/Specifications/OpenSearch/1.1

Conley, D. (2008). *Elsewhere, U.S.A.* (First Edition ed.). New York: Pantheon Books.

Cutts, M., & Shellen, J. (2005, January 18). *Preventing comment spam*. Retrieved April 22, 2012, from Official Google Blog: http://googleblog.blogspot.com/2005/01/preventing-comment-spam.html

de Valk, J. (2007, October 25). *Semantic HTML and Search Engine Optimization*. Retrieved April 21, 2012, from DEV.OPERA: http://dev.opera.com/articles/view/semantic-html-and-search-engine-optimiza/

Dracoulis, G. P. (2011). *Regional Social Network to Promote and Support Underrepresented Minority Scientists.* Worcester Polytechnic Institute, Computer Science. Worcester: Worcester Polytechnic Institute.

Fekete, G. (2008, December 4). *Designing The Holy Search Box: Examples And Best Practices*. Retrieved April 19, 2012, from Smashing Magazine:

http://www.smashingmagazine.com/2008/12/04/designing-the-holy-search-box-examples-and-best-practices/

Guha, R. (2011, June 2). *Introducing schema.org: Search engines come together for a richer web*. Retrieved February 18, 2012, from Official Google Blog: http://googleblog.blogspot.com/2011/06/introducing-schemaorg-search-engines.html

Hansson, O. (2011, June 7). *Authorship markup and web search*. Retrieved March 15, 2012, from Official Google Webmaster Central Blog: http://googlewebmastercentral.blogspot.com/2011/06/authorship-markup-and-web-search.html

Iskold, A. (2007, January 16). *The Art, Science and Business of Recommendation Engines*. Retrieved April 11, 2012, from ReadWriteWeb: http://www.readwriteweb.com/archives/recommendation_engines.php

Janrain Inc. *Five Best Practices in Online User Registration.* Janrain Inc.

LectureBank. (2011, August 1). *Welcome to LectureBank*. Retrieved January 1, 2012, from LectureBank: http://www.lecturebank.org

Marshall, M. (2006, December 10). *Aggregate Knowledge raises $5m from Kleiner, on a roll*. Retrieved April 12, 2012, from VentureBeat: http://venturebeat.com/2006/12/10/aggregate-knowledge-raises-5m-from-kleiner-on-a-roll/

Matthijs, N. (2008, March 3). *HTML, the Foundation of the Web*. Retrieved April 21, 2012, from Web Page Design for Designers: http://www.wpdfd.com/issues/86/html_the_foundation_of_the_web/

Mill, J. S. (1909). *On International Trade* (7th Edition ed., Vol. III). (W. J. Ashley, Ed.) London: Longmans, Green and Co.

Nelson, L. (2009, September 22). *Minority Students Needed in Math and Science to Combat 'Brain Drain,' Professors Say*. Retrieved October 14, 2010, from The Chronicle of Higher Education: http://chronicle.com/article/Minority-Students-Needed-in/48568/

Netflix, Inc. (2006, October 2). *Frequently Asked Questions*. Retrieved April 14, 2012, from Netflix Prize: http://www.netflixprize.com/faq

Netflix, Inc. (2006, October 2). *The Netflix Prize Rules*. Retrieved April 14, 2012, from Netflix Prize: http://www.netflixprize.com/rules

Nielsen, J. (2001, May 13). *Search: Visible and Simple*. Retrieved April 19, 2012, from Alertbox: Current Issues in Web Usability: http://www.useit.com/alertbox/20010513.html

Pariser, E. (2011). *The Filter Bubble* (Kindle Edition ed.). New York: The Penguin Press.

Pilgrim, M. (2010, August 24). *"Distributed," "Extensibility," & Other Fancy Words.* Retrieved February 16, 2012, from Dive Into HTML5: http://diveintohtml5.info/extensibility.html

Pilgrim, M. (2009, April 17). *The Road to HTML 5: Link Relations*. Retrieved April 22, 2012, from The WHATWG Blog: http://blog.whatwg.org/the-road-to-html-5-link-relations

Pilgrim, M. (2010, August 24). *What Does It All Mean?* Retrieved January 26, 2012, from Dive Into HTML5: http://diveintohtml5.info/semantics.html

Taxonomy Strategies. (2004, February 10). *What are the best practices in web site search?* Retrieved April 19, 2012, from American Society for Information Science and Technology: http://www.asis.org/Chapters/asispvc/feb_10_2004/Web_Search_Best_Practices.pdf

Appendix A: SVN Deploy vs. FTP Upload

When deploying a site, designers and developers have a number of options. Each of these has its own advantages and disadvantages. In this document, we will take a closer look at two of the popular options, File Transfer Protocol (FTP) upload and Subversion, and how they function in the context of a dynamic site with multiple stakeholders. Furthermore, we will examine approaches to deployment using combinations of both and make a recommendation for best practices.

File Transfer Protocol

FTP's main advantages are its simplicity and ubiquity. It is the most common way to transfer files between a local and remote server, and there are numerous tools available to accomplish just that. Virtually all web servers are outfitted to work with FTP by default, and information, documentation, and support for using FTP is often readily available no matter what web host you are using. When working on a shared server, FTP may be your only option since it does not require shell access.

Subversion

Subversion, or SVN, is a popular Software Configuration Management (SCM) system. It allows users to store information about different versions of source files for a particular project, including who made which changes in a multi-developer environment. FTP stores no such information-the only readily accessible information that might be pertinent to versioning would be the date and time the last upload operation was completed. It requires an extra step in that it forces developers to "commit" any changes first to a repository server, and then to update the corresponding files on a live version of the site when they are ready. It also requires the separate configuration of a Subversion repository server,

which may or may not be readily available (either from a developer's institution or the company providing web hosting). Shell access is also necessary, although it is often only available on higher-end dedicated or semi-private servers/hosting packages. This additional overhead also increases technical complexity to some degree. Finally, Subversion clients for developers working across platforms can be a bit harder to come by (although one usually comes preinstalled for use through the terminal on Mac/Linux/UNIX systems).

SVN+FTP

One might choose to use FTP or Subversion exclusively, in which case the pros and cons mentioned above would suffice. However, developers might choose to use a combination of the two technologies-a Subversion repository for version control and perhaps for backup, with someone designated to upload files via FTP when they are ready for deployment. This, however, negates some of the benefits of using a version control system in the first place-being able to carry out operations to compare versions in the repository vs. those deployed on the live server and being able to roll back to previous versions quickly and easily, for example.

Recommendations

If your institution or hosting company has a facility to create Subversion repositories and your hosting company allows shell access (via SSH) as part of your hosting package, *use Subversion exclusively even if you are a solo developer*. If both of those conditions are not met, it makes more sense to use FTP or consider upgrading. MediaTemple (http://www.mediatemple.net) is a good and inexpensive hosting provider that provides shell access and a Subversion server as part of their standard packages. It makes sense to

use a version control system if it is available and might fit into your workflow, since you never have to worry about losing code. If something breaks, you can identify who might have committed the offending changes and when they might have been introduced, and quickly roll back. With FTP, you may have already overwritten the only copy of the older files-the ones on the server. With Subversion, each time a change is made to your files, it is preserved and trackable forever. In addition, Subversion eases team collaboration and evaluation since everyone can contribute without fear of overwriting someone else's work and you can see who added what. Again: if you can use Subversion, do it.

Additional Reading

For the definitive manual on how to use Subversion:
Version Control with Subversion
http://svnbook.red-bean.com/

For additional information on why and how to use SVN for web projects:
"Subversion for Designers" by Chris Nagele
http://thinkvitamin.com/design/subversion-for-designers/

For more information about using Subversion to deploy PHP projects:
"PHP Deployment with Subversion" by Lorna Mitchell
http://www.slideshare.net/lornajane/php-deployment-with-svn

Appendix B: Using Subversion to Deploy to LectureBank

Step Zero: Request Access to the Repository and Check Out a Fresh Copy

Go to http://sourceforge.wpi.edu and create an account if you haven't already. Contact whoever is currently administering the repository and request that they add you as a member of the project. Once you've been added, open up your favorite SVN tool or a terminal window. Navigate to the directory where you want to check a copy out to. If you're using a terminal, copy and paste the checkout command made available to you under the "Source Code" tab of the project (if you want to check out just the files into the current working directory, add a space and a period " ." to the end of that command). Enter your password when prompted to check out a fresh copy of the repository.

Step One: Update Your Copy

Once you've already checked out from the repository, update your current working copy by navigating to the directory to which you initially checked out, then typing "svn update" and providing your password.

Step Two: Make Your Changes

Make any edits, additions, deletions, or changes to the files and their structure. Save.

Step Three: Update Again

Remember, before you make any commits, always update. If you think there might be any changes you will have to resolve, type "svn diff" to show which files have changed. Test and make sure everything still works. If it doesn't, make it.

Step Four: Add or Delete Files From the Repository

Use "svn add" or "svn delete" as appropriate, based on the results of "svn diff"

Step Five: Commit

Once you're sure all collisions are resolved and everything is working, type "svn commit" into the terminal window and wait for the operation to complete.

Step Six: Deploy to Staging

SSH into the MediaTemple servers with [redacted]. At the shell prompt, type "./stagedeploy" to deploy to staging.lecturebank.org. [redacted]. If something breaks, navigate to ~/domains/staging.lecturebank.org/html and type "svn log | more" and look for the previously checked-out revision number, then type "svn up –r ####" where #### is the pertinent number. This will restore the previously checked-out version until you fix the problem and redeploy.

Step Seven: Deploy to Live

SSH into the MediaTemple servers and type "./livedeploy" and everything should be updated on the live server. *DO NOT* do this until you have tested on staging.

Appendix C: Speaker Use Case

A postdoctoral researcher or other graduate/postgraduate student-perhaps even a faculty member-might want to put him or herself into the system in order to find talks nearby to participate in. The following flow details the steps he or she might take to meet that goal.

1) Sign Up for LectureBank
    a. Navigate to "lecturebank.org" or find LectureBank via another organization's website or through a search engine
    b. Find the "Sign Up" link in the login control panel at the top left, or in the large button at the bottom of the front page
    c. Complete the "One Step Signup" process by picking a username, entering an email address, and choosing/confirming a password. All validation is done on the same page here-usernames and email addresses are checked asynchronously for uniqueness, and passwords are checked for strength and whether they match.
    d. Check the email address they provided for a validation link delivered by LectureBank, then click it to validate their account
2) On First Run, Enter Basic Information
    a. The "Create Your Profile" prompt automatically presented can be skipped, but prompts for full name, home institution, zip code, primary field of study, and keywords for areas of interest. The home institution and areas of interest fields are autocompleting, and the zip code is checked against Google maps to convert it into latitude and longitude upon submission.

b. Scheduled Talks, Previous Engagements, and Research Publications will initially all say “None yet! Why not add one?” while linking to their respective add entry forms.

3) Run a search for talks to participate in, or wait for results to be delivered
    a. Click on “Speakers” to be taken to the search form for speakers to find matching planners and opportunities in their area.
    b. Enter information in the search form or, if enough information is already in the user’s profile, peruse the list of automatic results gathered based on the field of study, areas of interest, and ZIP code entered
    c. If location information is not already in the user’s profile, prompt for a ZIP code or an institution. The search form might prompt for a radius or just automatically sort by distance and cut off at 100 or so results. The user could refine this by only looking at talk/people results at a certain university, or perhaps posted by a certain other user.
    d. In the future, we may deliver some automatic suggestions for people who might be relevant to a user to his or her home screen or, if opted-in, inbox.

Appendix D: Planner Use Case

A talk organizer might want to submit a speaking opportunity for members of the site to review, or actively search for candidates. The following flow details the steps he or she might take to meet that goal.

1) Sign Up for LectureBank
    a. Navigate to "lecturebank.org" or find LectureBank via another organization's website or through a search engine
    b. Find the "Sign Up" link in the login control panel at the top left, or in the large button at the bottom of the front page
    c. Complete the "One Step Signup" process by picking a username, entering an email address, and choosing/confirming a password. All validation is done on the same page here-usernames and email addresses are checked asynchronously for uniqueness, and passwords are checked for strength and whether they match.
    d. Check the email address they provided for a validation link delivered by LectureBank, then click it to validate their account
2) Submit Opportunity
    a. Click the "Planners" tab in the header of any page
    b. Select the option to post an opportunity
    c. Fill in and submit the form asking for the name and location of the event, the sponsoring institution, the date for the opportunity to open and the date to close, the date(s) or available date ranges for the event, a further description of the event, a description of what the organizers are looking

for in a speaker, further instructions to apply to be considered for the event, and keywords.

d. Check under the “Planners” tab in the “Opportunities you’ve posted” section for responses to the opportunity and for suggested users that match the criteria, tags, or description entered.

3) Search for Candidates

a. Click the “Planners” tab in the header of any page

b. Enter some initial search terms into the search box presented and click “Search”

c. The user can choose to either click on a result to view more information, to input a new search, or to click on a displayed tag (as a cheap/naive "suggestion") and continue browsing

i. If the user clicks on a result, the system displays the selected user's profile and we continue

ii. If the user enters a new search or clicks on a tag, the result is the same: the system loops back to part c with the new term

d. If the user is satisfied with the result, he or she can choose to send a message using the email address displayed on the person’s profile, otherwise the user can turn back to the results page or execute a new search

Appendix E: Database Schema

Table structure for table events

Field	Type	Null	Default
id	int(11)	Yes	NULL
name	varchar(85)	Yes	NULL
inst_id	int(11)	Yes	NULL
desc	text	Yes	NULL

Table structure for table fields

Field	Type	Null	Default
code	varchar(3)	Yes	NULL
name	varchar(40)	Yes	NULL

Table structure for table institutions

Field	Type	Null	Default
id	int(11)	Yes	NULL
name	varchar(85)	Yes	NULL
address	varchar(50)	Yes	NULL
city	varchar(50)	Yes	NULL
state	varchar(2)	Yes	NULL
zip	int(5)	Yes	NULL
lat	decimal(10,6)	Yes	NULL
lon	decimal(10,6)	Yes	NULL

Table structure for table lectures

Field	Type	Null	Default
id	int(11)	Yes	NULL
title	varchar(140)	Yes	NULL
eventname	varchar(140)	Yes	NULL
loc_id	int(11)	Yes	NULL
abstract	text	Yes	NULL
link	varchar(255)	Yes	NULL
creator	int(11)	Yes	NULL
locdetail	varchar(255)	Yes	NULL
start	datetime	Yes	NULL
end	datetime	Yes	NULL
timezone	int(2)	Yes	-5
sequence	tinyint(4)	Yes	0
updated	timestamp	Yes	CURRENT_TIMESTAMP

Table structure for table lecturetags

Field	Type	Null	Default
lecture	int(11)	Yes	NULL
tag	int(11)	Yes	NULL

Table structure for table opportunities

Field	Type	Null	Default
id	int(11)	Yes	NULL
creator	int(11)	Yes	NULL
title	varchar(140)	Yes	NULL
eventname	varchar(140)	Yes	NULL
loc_id	int(11)	Yes	NULL
descrip	text	Yes	NULL
link	varchar(255)	Yes	NULL
locdetail	varchar(255)	Yes	NULL
start	datetime	Yes	NULL
end	datetime	Yes	NULL
instruct	text	Yes	NULL
open	date	Yes	NULL
close	date	Yes	NULL
timezone	int(11)	Yes	NULL
timestamp	timestamp	Yes	CURRENT_TIMESTAMP

Table structure for table opportunitydates

Field	Type	Null	Default
id	int(11)	Yes	NULL
opp_id	int(11)	Yes	NULL
start	datetime	Yes	NULL
end	datetime	Yes	NULL

Table structure for table opportunitytags

Field	Type	Null	Default
opportunity	int(11)	Yes	NULL
tag	int(11)	Yes	NULL

Table structure for table research

Field	Type	Null	Default
id	int(11)	Yes	NULL
uid	int(11)	Yes	NULL
title	varchar(140)	Yes	NULL
link	varchar(140)	Yes	NULL
yr	int(4)	Yes	NULL
abstract	text	Yes	NULL
journal	varchar(255)	Yes	NULL
volume	int(5)	Yes	NULL
issue	int(5)	Yes	NULL
startpg	int(5)	Yes	NULL
endpg	int(5)	Yes	NULL
doi	varchar(255)	Yes	NULL
updated	timestamp	Yes	CURRENT_TIMESTAMP

Table structure for table researchtags

Field	Type	Null	Default
research	int(11)	Yes	NULL
tag	int(11)	Yes	NULL

Table structure for table tags

Field	Type	Null	Default
id	int(11)	Yes	NULL
tag	varchar(50)	Yes	NULL

Table structure for table userfields

Field	Type	Null	Default
uid	int(11)	Yes	NULL
fcode	varchar(40)	Yes	NULL
type	varchar(5)	Yes	NULL

Table structure for table userinstitutions

Field	Type	Null	Default
uid	int(11)	Yes	NULL
instid	int(11)	Yes	NULL
type	varchar(5)	Yes	NULL

Table structure for table userinterests

Field	Type	Null	Default
uid	int(11)	Yes	NULL
intid	int(11)	Yes	NULL
type	varchar(5)	Yes	NULL

Table structure for table users

Field	Type	Null	Default
id_user	int(11)	Yes	NULL
username	varchar(20)	Yes	NULL
password	varchar(32)	Yes	NULL
email	varchar(64)	Yes	NULL
status	varchar(20)	Yes	NULL
activationkey	varchar(100)	Yes	NULL
name	varchar(50)	Yes	NULL
zip	int(5)	Yes	NULL
updated	timestamp	Yes	CURRENT_TIMESTAMP

Table structure for table ziplocation

Field	Type	Null	Default
zip	int(5)	Yes	NULL
city	varchar(50)	Yes	NULL
state	varchar(2)	Yes	NULL
lat	decimal(10,6)	Yes	NULL
lon	decimal(10,6)	Yes	NULL

Appendix F: Selected SQL Queries

Ordering the most active users

```
SELECT COUNT(*) AS count, uid, users.name, users.username
FROM (SELECT uid AS uid FROM research UNION ALL SELECT
creator AS uid FROM lectures) AS useractivity, users WHERE
useractivity.uid = users.id_user GROUP BY uid ORDER BY
count DESC LIMIT 10
```

Ordering the most recently active users

```
SELECT * FROM (SELECT * FROM (SELECT id_user AS uid,
updated FROM users UNION ALL SELECT creator AS uid, updated
FROM lectures UNION ALL SELECT uid, updated FROM research)
AS cat ORDER BY updated DESC) AS ordered GROUP BY uid ORDER
BY updated DESC LIMIT 10
```

Random related opportunities (like you)

```
SELECT opp.*, users.name AS creator, users.username AS
username, institutions.name AS location FROM (SELECT id
FROM (SELECT intid AS tag FROM userinterests WHERE uid =
'$uid' UNION ALL SELECT researchtags.tag AS tag FROM
researchtags, research WHERE research.uid = '$uid' AND
researchtags.research = research.id UNION ALL SELECT
lecturetags.tag AS tag FROM lecturetags, lectures WHERE
lectures.creator = '$uid' AND lecturetags.lecture =
lectures.id) AS usertags NATURAL JOIN (SELECT tag,
opportunity AS id FROM opportunitytags) AS opptags GROUP BY
id ORDER BY RAND() LIMIT 10) AS randopps NATURAL JOIN
opportunities AS opp, users, institutions WHERE
users.id_user = opp.creator AND institutions.id =
opp.loc_id AND (opp.open <= CURDATE()) AND ((opp.close >=
CURDATE()) OR (ISNULL(opp.close) AND (opp.start >=
CURDATE())))
```

Appendix G: System Diagram and Flow