

**Transcript of a conference call on enterprise directory and
metadirectory issues
December 13, 2001,**

Interviewers:

Richard Jones, University of Colorado/Internet2

Keith-Keith Hazelton, University of Wisconsin, Madison

Participants:

Todd-Todd Pickett, Michigan Technological University

Ben-Ben Oshrin, Columbia University

Frank-Frank Grewe, University of Minnesota

Staff Support:

Lisa-Lisa Hogeboom, Internet2

Ellen – Ellen Vaughan, Internet2

Richard So we're all here. That's great; I'm Richard Jones at the University of Colorado.

Richard Let me thank all of you for agreeing to do this, especially on such short notice at this time of year. We really appreciate it. Let me tell you what our thinking was behind scheduling the two calls. (This is the second call.) We want to do two things hopefully. One is to get you to tell the story of your enterprise directory effort and especially anything to do with meta-directories or linking directories together. We're recording this call and we're going to produce a transcript, and we think with just a little bit of editing we can put it up on the Web as a background document for institutions who have not yet started or are just thinking of starting a directory effort and they don't know the questions to ask, they don't know the problems they're going to have to solve, and they just need some orienting or background information. We think that the two documents from these two calls will be very helpful to those people. And then secondly, we want to mine it ourselves looking for what we're called gems of wisdom, because we'd like to produce a shorter document that's sort of analogous, a companion, to the LDAP recipe that captures best practices or strong do's and strong don'ts from people who've actually done the work. So those are the two goals that we have in mind for these calls.

We sent you an outline of what we wanted to cover, and let me go over that and remind you and make sure that everybody understands or answer any questions about that. And what we thought we'd do in terms of a format is we would go around each institution, that person can tell the story, and then at the end others can ask questions or make comments and then we'll move on through all three. There were six basic areas that we wanted to hear about: (1) the goals of the project (sort of a general description); (2) the background (the campus environment you started with, the data sources or systems), and maybe data owners that you had to deal with political issues like that; (3) the major problems you had to solve to accomplish the project; (4) the methods you used (software platforms, experiences with those platforms, data transformation issues that you had to deal with); (5) successes and failures that you want to highlight; (6) advice to others that are going to do this same sort of thing. And we think that the three really meaty ones of the six are the background, your campus environment, the major problems that you had to solve and the methods that you used to get there. And we're especially interested in hearing problems or thoughts or things that you did in certain areas that have to do with linking directories or meta-directory issues, any legacy interactions that you had to deal with, problems with correlating identifiers and how you dealt with that, pushing data back into legacy systems, whether you are doing that or want to, data cleansing and reconciliation problems and how you address those, linking with other directories, any multi-campus issues that you might have, and interacting with local or departmental directories and person directories like LANs and maybe address books and things like that. So those are the topics and things we'd like to hear from you. I think first we should let everybody introduce themselves and the institution, and then we'll start out and hear from everybody. Keith, do you want to add anything first?

Keith No, just again my thanks and appreciation for people being able to do this on really short notice. I'm looking forward to hearing the discussion. I'm Keith Hazelton, the University of Wisconsin, Madison. IT Architect and the director of the MACE-Dir working group. So this is great stuff as far as I'm concerned.

Richard Ben, do you want to introduce yourself first?

Ben Sure, I'm Ben Oshrin from Columbia University. I am a senior programmer within the systems group. The name you've probably heard from us is Alan Crosswell. I work under his group.

Richard Okay, thanks. Frank.

Frank I'm Frank Grewe, the University of Minnesota, I'm the manger of Internet services.

Richard Okay, and Todd.

Todd I'm Todd Piket at Michigan Tech. I'm an analyst programmer in distributing computer services. I have lots of duties but one of my main duties is to maintain the directory services here.

Richard Okay, thanks. Well, if anybody is dying to go first, speak up, otherwise we'll just go in the order--

Todd Actually I am, because I have to leave very, very soon.

Richard Okay, good, well, Todd, why don't you start and tell us about Michigan Tech.

Todd Well, when I first got here I was actually hired because I had directory server experience, specifically with iPlanet. So we use the iPlanet directory server exclusively in a Unix environment. It started out on Solaris, we recently moved it to a couple of Linux boxes as well. But the main goal was to sort of create a white page enterprise directory just for look-ups and replace the legacy ph system that we currently had with an LDAP directory server. That was the first goal. So we did that using--we use SCT Banner here for our student data and faculty and all that. So using Oracle, we created a table that contains--the Oracle table is sort of my meta-directory between Banner and my directory server. So I took all the disparate sources that I needed from Banner and put them into this one table and then I take that table and turn it into LDIF, and then I dump the directory server data into LDIF and I essentially do a diff on that to figure out what needs to be changed, and that runs on a nightly system, and that's a process that I wrote using Perl pretty much exclusively to do the diff-ing, to find the deltas, and there's some PL/SQL in there too that actually builds the intermediary table.

Most of that was already in place when I got here, because that's pretty much what they did with ph, I had to do a few other things and I had to build some object classes and stuff for our directory server, but that's all done now. And the way we replaced ph, we didn't actually cause a problem by removing the ph protocol, so you can still type ph and use it, but we use Qualcomm's ph-to-LDAP adaptor so if you make a ph request and it goes to the ph-to-LDAP adapter and turns it into an LDAP request and returns information just as ph

would. I actually rewrote a lot of their ph-to-LDAP adaptor, so it's kind of Michigan Tech's now. If anybody wants it let me know!

We are now getting into some authentication things with that with the directory server. I wrote a Web single sign-on system that does a little bit of authorization as well using the LDAP module with Apache. So I also replaced our--we have something here called MichNet, Merit Networks. I don't know if you ever heard of that, but it's a free dial-in service provided throughout Michigan for education. Source students can dial in wherever they are in Michigan pretty much with a local phone call, but they're authenticated against us so that they can get to the university resources here. But the authentication mechanism that was being used was Radius, and then on the backend up here, our Radius was actually talking to Kerberos and I replaced that with Radius on the backend speaking to the directory server. So now people use the Web single sign-on system to log in with the directory server and they use Radius--or when they dial in, they also log in with the same password. So what we're trying to do is get it into a single password environment for authentication using the directory server as well. And that actually involved integrating another legacy system called the NID, which is the Network Information Database, which is where we keep all of our users and all of their Unix user ID's. We're almost a strictly Unix environment here, at least on the backbone, so all the PC's in the labs and everything talk to samba servers, so everybody has a Unix uid. And that's how our unique identification really works. And SCT Banner people also have a "pinum," which also serves as a unique identifier. The pinum from Banner is what I actually used to build a person distinguished name in the directory server so that I know that they're never going to move because the pinum never goes away, it's never reused and it's not recycled.

Some other things we've been experimenting with in authentication is using pluggable authentication mechanisms to authenticate against the directory server that we can maybe get departments to start using this campus password idea, and so we can really get into a single password environment. And maybe from there jump into PKI.

But in terms of major problems to solve, when we first started moving the ph things into the directory server I discovered that we had 32

different addresses in SCT Banner for a single person, and I had to figure out--well speaking with the data custodians in the registrar's office and human resources, I had to figure out which address field I really wanted, which address codes I could use, which ones were the cleanest, all of those sort of things. I don't do any data scrubbing, I leave that up to them, so basically I just take the data and put it somewhere else. So if it's messed up it's really not my problem. But that was a big deal. That took a good couple of weeks. And then, of course, everybody was concerned as to why I wanted this data, who's authority was it that I was using it, are we going to change it at all, are we going to allow them to change their address and have it right back to the database. We don't do that right now, we would like to, but because of all the political issues that that gets into, we don't do that yet.

So far it's been a very successful--I don't know, am I leaving anything out? Ours is a pretty simple environment, we don't have multi-campus issues. We're just beginning to start pushing data back into some legacy systems using the email address and only with a small subset of people. And that's actually with our distance education groups, our distance education group of people because they're a smaller subset and it's really not going to affect the student body that's here in any way. So fortunately, like I said, we have--the identifier issue had already been solved for me. User ID's are unique across the university. That's what the Network Information Database is for (the NID), that's why it was created was to make sure that all the user ID's were always the same and all the Unix user ID's were always the same, so that if a person is in multiple departments, taking classes in multiple departments, their user ID and their Unix user ID number is always the same wherever they go. That's pretty much it.

Richard Any advice to people just starting the process?

Todd Start small and don't anticipate that it's going to take a month. That's the biggest thing. Most people seem to think that it's not going to take very long, and it took us like a year to replace the ph system. I thought it was going to take maybe a couple of months, but once all the political stuff started to hit the fan--

Keith On the NID stuff, again, you say that was there for you, but do you know how they go about reconciling that entity, say, if someone comes in first as a student and then they--oh, they're still a student and

become an employee, I assume that it's going to be one feed from the student information systems and one from the HR.

Todd Right, okay. That's a good question. That's exactly why the NID is separate from SCT Banner. In the NID the only thing that's kept is the person's user ID, like their login ID, and their Unix uid number. The department and the pinum and the employee--their personal status, whether they're a student, an employee, or a faculty member, or staff of faculty, whatever, that's all kept in SCT Banner. And so the feed process that I have gets all of the status codes and everything out of Banner and then gets all of their log-in ID's and everything from the NID, and I think they're linked to the pinum.

Keith And pinum--

Todd So the pinum is really the unique identifier.

Keith And can you talk a little more about where that comes from and what it is?

Todd The pinum is part of SCT Banner. When a person is first entered into the Banner system, they are given this thing called a pinum, personal identification something. And that is their pinum until the end of time. Even if they get deleted from the system, that pinum will never get reused.

Keith Some of these questions are just based on my lack of knowledge of Banner.

Todd Right, no, that's fine.

Keith Like, that is the place in your system where the identify management function is getting done.

Todd The pinum is being used as a mapping. I personally don't believe it should be because if we ever switch from the Banner system, we're kind of hosed. So what I would like to do is create a new number, call it whatever you want, Michigan Tech unique identifier, and that unique identifier will be the person's distinguished name, identifier, and all that, and from that unique identifier you can get their pinum, their user ID, their Unix user ID, whatever. So like a mapping would be created.

Keith Do you have yet any developers or systems that are saying "we'd like to get an extract from the directory and refresh that every once in a while?"

Todd And extract to do what?

Keith Well, we've got people that at the very least let's say they'd like to extract the identifiers so they can go back and look in the directory for things like email address, where that's the authoritative

source, but they want some kind of local store where they carry that identifier as a kind of a foreign key and then add their own app-specific things like what their profile is or the last time you logged on or something like that. This is a general affiliated directory problem where you want to create some kind of persistent link between person entries and distinct directories and I'm just wondering if you're getting people wanting to do that.

Todd Not yet. I can see the wheels starting to turn in people's heads, but they haven't quite gotten that far yet. So not yet.

Keith Enjoy it while--!

Todd Yeah, I am, believe me, I am. When I was at Ford, that happened a lot, and we never really had a way to solve that, so--

Keith Because for example, I assume pinums can in fact change if they notice for example that somebody's in there twice or whatever, one of those pinums might disappear.

Todd Yes, actually--

Keith And again, it works perfectly fine within the confines of its own system but if other systems are relying on that thing to be persistent over time and they're building up--storing their information under a pinum and it changes out from under them, they're kind of left--

Todd Right, that's actually exactly right. When I came back to the University as staff member, because I left when I graduated, I left and then I came back as a staff member, but they created a new pinum for me, even though I still already existed in the Banner system as a student. And fortunately I knew how it worked so I caught it and said, "Hey, I'm still here!" But occasionally that does occur, there are two pinums for a single person, but the NID, we'll usually catch that because there's a primary key based on user ID and pinum, and so the way that the user ID is generated almost always catches that, a dual pinum scenario.

Keith So you've got some universal process even up to and including distance education people where they--as part of getting associated with the University they go through some account activation things to get their log-in ID in, and that's when you're doing all these checks, right?

Todd Yep.

Keith Essentially every person funnels through that, some account activation or creation mechanism.

Todd Right, everybody goes through Banner because you can't get a login ID until you're in Banner because you have to have a pinum

first. And so you go through Banner and you exist there, and then whoever essentially owns you as a student then creates an account for you in the NID and it generates your user ID for you and the NID makes sure that it's unique within it's own little subsystem. And then my feed process, which runs at night, will pick you up as a new add and add you to the directory. And that's pretty much how it works.

Richard Ben, Frank, do you have any comments or questions?

Ben It sounds like you've run into a lot of sort of similar situations to what we have. I won't go into my official term, but I think you are not alone!

Frank Yes, I definitely ditto that.

Richard Now do you run Active Directory or anything like that on the campus?

Todd Me? At Michigan Tech?

Richard Well, does anybody?

Todd We do not. Like I said, we're almost a 95 percent Unix shop, including in the labs. Though there's been some leaning towards Windows and Active Directory, but it hasn't been an issue yet.

Ben We're an emphatic no on Active Directory, we're really big on open-source solutions, solutions that we can fix the code when we discover about it.

Richard Okay, anymore questions or comments for Todd? I'm not hearing anybody.

Todd Sounds like a no to me, says Todd.

Richard Alright, well thank you. You said you're going to have to leave pretty soon?

Todd Yeah. Actually I'm already late so I really have to leave right now.

Richard Okay.

Todd I would like to hear other people's things. Can I get a copy of the transcription when I'm gone?

Richard Of course.

Todd Okay, great.

Richard Thank you very much for participating today.

Todd Thank you for inviting me.

Richard Okay, bye.

Todd Bye, bye.

Richard Well, now we have Ben and Frank, right? Are you still there?

Ben Still here.

Frank Yep.

Richard Well, Ben, do you want to go next?

Ben Sure. I've jotted down a few notes but I'm sure I'll leave something out. I guess, let's start with the general description. Our directory infrastructure, the LDAP portion of it, has sort of grown organically off of what was preexisting. The preexisting stuff like Todd was describing, was ph-based or qi-based, and I'll get into the details of sort of how the data got assembled for that, because it still applies to how our directory stuff for LDAP works. For clarification, ph is the frontend and qi the backend.

But basically in a sense what we ended up doing about four years ago or so was switching the publishing mechanism from qi to LDAP, which was obviously a very visible thing for pretty much everybody who uses directory services here, but from a technical standpoint and how we were sort of assembling the data and massaging it, it didn't actually change that much. The campus environment is we've got about 30,000 students and another 15,000 faculty, then when you throw in staff and alumni, the total number of people that we'd be concerned with is on the order of a quarter of a million. It's a very decentralized campus in many aspects. The data comes from over 80 different locations, so obviously some of them are significantly bigger than others. For example, the student source will be much larger than one small department, which feeds us five people. The systems that it comes from we now sort of fall into two categories, the batch systems where the entity that's providing us the data has some--has their own database that they use to manage their SCT of people and then they essentially feed us an extract so that applies to students and staff and faculty and so on. And then there's the smaller organizations, say, a department that feeds us ten people--we've been pushing them towards using a Web interface and I'll get more into that momentarily.

The data is owned by the people who give it to us and we hold that very strictly. If somebody needs to have their data modified because their name is misspelled or something, we universally send them back to the source. We will not fix the data for them unless it's a priority change like somebody is being harassed or whatever, and we need to make them disappear electronically. The most common cases are name changes (marriage, divorce, etc) and bad auto-generated Ids (ass22, etc.)

Let's see, so probably our biggest major problem (and this is an ongoing problem, which is extremely difficult to deal with) is that we get bad data from people. Our Web-based feeder for the small departments has been a major success in dealing with that because we put in a lot of checking before they can give us the data. You know, is the birthday a four-digit year, or is the month less than 13, has actually saved us from a lot of problems. It's harder to do that sort of data on the bigger system feeds, and so we still get some bad data coming in from there that we either have to resolve manually or we have to send a note back to whoever gave it to us to have them do it, but bad data is probably one of the more significant problems that we still encounter. Probably second to that is a bad database! We're still running in Ingres, I think we've got OpenIngres Version II 2.0, and every now and then we come up with one of those bugs that the tech support people don't really know how to solve, so we've been evaluating other databases, but again, that's an externally difficult thing to sort of pull out underneath everything. Where the database comes into play is all the data that comes from all the various feeds is chewed on overnight and using a variety of locally-written programs that have been developed and modified over the years, takes the data from the various sources and puts it into a standardized format into the database, does whatever relevant manipulations over and including things like assigning affiliations to people and then sort of linking people up. Course affiliations have also been extremely useful in our deployment of CourseWorks, aka Prometheus. CourseWorks is able to use this data to automatically determine who is affiliated with what classes, so little if any intervention is required on the part of the faculty or their assistants.

I'll get back to the linking people up in a minute. I'm just going down my notes, so I'm trying to--that's why this gets a little disjointed from time to time. In terms of moving--to stick with the major problems to solve, in terms of moving things from qi to LDAP, that was actually a pretty straightforward move. We had some minor problems with the qi data being formatted specially for qi, using things like stuffing three titles into one title field and separating them by commas because qi didn't have the concept of multiple titles, which we cleaned up for LDAP, which does. But other than that, that move went pretty simply. Qi also uses magic cookie authentication. Essentially you had a special password to authenticate to the database and we moved

that to a concept of a set of super users who have publishing access to the LDAP directory, Kerberos based, and that ranges from end users being able to see their own stuff that normally isn't published publicly to staff being able to see anybody's information for diagnostic purposes. That is, the directory server has a list of authorized groups of users and what they have access to. Members within those groups bind to the LDAP server using a Kerberos ticket to authenticate themselves.

I guess one last major problem that we still have is that nothing is really static. There's not really any permanent unique identifier, although we try to make there be one. Social security numbers are not necessarily guaranteed to be unique. Sometimes we get phony social security numbers from departments when in fact the person comes from somewhere else and has a real social security number. We have a mechanism for sort of unifying all that's stated the best that we can and assigning University Network IDs which are just Kerberos handles, which are as unique as we can get, but even those occasionally change when somebody is getting harassing email and needs their identity to switch or something like that. So I guess there are ways we can sort of deal with this, but there's not one clean method to permanent identify a person ever.

The way we try to match people coming from different feeds as a unique person is essentially on what might be considered a total point system where we look at the data that comes in from the various feeds and if enough of it looks like a good match then we consider that the same person. And this generally works like a social security number is worth a certain value, maybe the name is worth another value and then once you pass a certain threshold we assume that multiple entries are the same person. If they're not, they need to be manually resolved or if we miss somebody who is, either the feed that we're getting needs to be corrected so that it has the appropriate information in it or we need to do some other sort of manual reconciliation. The first time a person shows up in a feed, in any feed, that we determine their unique individual, we assigned them the kerberos handle and then that sticks with them theoretically forever, but as said, even that can change.

This all happens during a nightly batch job. When the batch job is finished--well when that portion of it is finished we do a database extract which generates an LDIF file. And we also output some other files for entrusted entities, like we generate a telecom extract for their purposes, which is just the same data or a subset of the data and essentially the same or similar format.

The LDAP database is rebuilt nightly from scratch. It's a full LDIF feed file. Any changes that were made manually during the day get blown away over night, assuming the batch job successfully runs. And let's see, so I guess overall this has been pretty successful. We've got pretty much everybody in the directory now. There are, like I said, about 300,000 people in it, including alumni. Attached to these people we have all sorts of--whatever information we are provided or we can sort of infer, including obvious things like their phone numbers and addresses, but also things like their affiliations, so where they come from, what privileges they get from that. We have course information in there, so if you're a student, the courses that you are enrolled in are attached to your identify, though not publicly visible, but it's accessible to the authorized entities. If you're a faculty, whatever courses you teach are also put in there. Pretty much anything that wants access to this information knows how to talk LDAP or knows how to talk to a broker that knows how to talk LDAP. In addition to generally also needing to speak Kerberos to do the initial authorization to access the information. And overall it works pretty well. Web authentication/authorization works essentially transparently to the user. Our Unix account creation system is tied into this also to see if you're eligible to create an account for example. It goes through a very strong authorization facility so we can literally have Web pages that are restricted to students enrolled in a class and the instructor and not really have to do any fiddling to get it to work.

The down side of the overnight batch loading is that priority changes are hard, so if we do have a case of student who's being harassed or staff who got fired under less than pleasant circumstances, then it's usually up to our group or the help desk group to stay on top of that change, which is usually done manually during the day, until the underlying feed is changed, so that the data correction shows up. So usually that means for a day or two somebody has to remember to keep taking this person out of the directory when they keep showing

up every morning. That puts a little bit of burden on the help desk. Also the overnight updates puts another little bit of burden on our help desk. If they tell somebody to go make a change and then the change doesn't take right away, they might get confused or it might become a little bit harder for somebody to diagnose what a problem is, what it looks like the underlying data and the database is correct, but it hasn't been reflected yet. We're working long-term to address this to move to real-time updates, but like changing the database out from underneath it, that's a pretty difficult challenge, well, at least within our model where everything is currently overnight batched.

As for advice to others, data is only as good as its sources (garbage in/garbage out), a lot of problems just come from bad data we get from people, and as I mentioned before, the Web-based feed editor was a major win in reducing that. If you're starting from scratch, good planning is essential. You want to know not only what you're going to need immediately, but what you might need down the road. You really want to do it right the first time because if you mess anything up it's going to take you forever to change it, especially as more and more people start using your services. You're then going to have to deal with either backwards compatibility issues or transitional issues, and it'll just make---the change that you thought, "Oh, it'll take five minutes to do later," is actually going to take five months to do because you have to deal with all of these secondary considerations.

We don't really have issues with linking with other directories because we really are mostly *the* directory. We do have some internal directories that we use that we have to deal with, but by and large it's not really an issues because we control them so we can fix that as we need to.

Multi-campus issues don't necessarily come up for us, even though I guess technically we do have a couple of different campuses. Anybody who's providing us data just shows up as another feed whether they're an affiliated institution like Barnard College or whether they're the major staff system. Then I guess to finish it up, interacting with local and personal directories. As I mentioned before, we don't really like the idea of closed-source directories, so we use--currently we're using open LDAP, we've got the slightly older version of the server running but it's been extremely useful to have

open source for this. We found a couple of bugs in the server where people were not showing up for some reason, we just spent half a day hacking through the code, found it, sent the change back to OpenLDAP, and the problem was fixed.

We have looked at making the directory available in some way, shape, or form for address books. For Netscape I think it just works if you tell it to use our directory. The results of the discussion of putting roaming information, the last time we looked at that, we were not happy with the security measures that were required to implement it, so we put it off and that really hasn't come up in quite a while. I think that's what I have on my list, so--

Richard Well good, thanks! Frank, Keith, questions or comments?

Keith I do have one. When you're linking people together from these 40 or 50 sources, once you've established that these two people really are the same, does that information--what should I say? Does that persist between creations and recreations of the directory or is that redone on the fly each time?

Ben It's essentially persistent, assuming you stay in the feeds. Once you're in the feed--basically one of the first things that happens is we look at the last feed and the current feed and figure out what's changed, and then we process that information rather than everybody's feed from scratch each time.

Keith Alright, well let's just take a hypothetical--people came in from say the student system and the HR system and they've matched, and then by accident on some particular day the student's side went in and it messed up the SSN. I don't know if that's one of the things you use to match. But messed up one of the values that you use to do the match between the things. When you recreate that night, what happens? How does that play-out?

Ben Right, that actually does occasionally happen, though I don't know about in that specific scenario.

Keith Right, that kind of problem.

Ben Yeah. And what will happen is in the case where you're in two feeds, your identity will remain because the other feed presumably is unaffected.

Keith What about the link between--

Ben Essentially what'll happen is you'll basically show up as a second user when the broken social security number shows up. We can then manually resolve that by what's best described as a hack. If

your data is corrected in the feed that was broken, you should automatically link back up once that happens, but you may have that second instance lying around with the wrong social security number, which may not affect you as a person but we may notice it on the systems end when we see that your name matches up to these two identifiers.

Keith Right. So that's the general philosophical thing that we have here too, which is that we don't--and I was kind of implicit in Todd's thing too--we don't get into the business of fixing people's bad data in the LDAP directory. We essentially push notice of bad data back to the source systems and since they're where the stuff is coming from, it seems like that's the right place that it should get fixed there and not get into the business of, "Oh yeah, we know that's the HR system, they always screw this up so we'll fix it again." But it sounds like that--anyway, it seems like we're all agreeing to go that route. Let them know about it and have them fix it.

Ben Right. It's pretty exceptional that we go in and actually muck in people's data. By and large we tell them to go back and deal with the source.

Keith Right.

Richard Ben, do you use the directory to deal with mail?

Ben Partly. Is there a specific sense that you're interested in.

Richard Well I'm just curious. Then what part do you--

Ben We do have sort of--well originally we just had a mailing address published with the user, and then somehow it became determined that we wanted both a published address and a delivery address. What happens now is a process:

1. Message arrives at columbia.edu gateway.
2. Message is examined for rejection characteristics (spam, invalid IP address, etc).
3. To: address is looked for in local sendmail aliases files (for things like mailing lists).
4. If #3 does not match, To: address is matched against valid unix users. This uses the standard getpwnam (see item #12, below) API, which calls our non-public LDAP server that maintains all unix account information.
5. If #4 does not match, "lookup by name" tries to fuzzy match the To: address against someone in the directory. If a unique match is found, the "mail:" attribute determines where the message goes.
6. If #5 fails, the message bounces.

In addition to "mail:", we also have a "mailedelivery:" attribute in our directory. This manifests itself as one of the aliases files in step #3.

Richard So you don't actually assign a first--an exact First.Last to a person, you deduce it on the fly with some fuzzy logic, is that what you're saying?

Ben Yeah, the way it originally developed was essentially with ph, that is the string you typed matched you uniquely in ph, the mail was delivered, otherwise you were given a message back that said who the matches were, as if you had just typed ph blah, blah, and it gave you an answer. Because there were people depending on that functionality when we did the switchover to LDAP, we ended up having to essentially re-implement it. We do have, again, our own ph-to-LDAP gateway that we wrote, but we also ended up writing a separate program that sort of emulates ph functionality directly without having the overhead of going through the gateway.

Richard Okay. And you said that you--in the directory, for a person, you know what course they're enrolled in or what courses they're teaching, and do you do that with attributes or groups?

Ben Basically we have an overloaded term in our not-very-well-formalized scheme I called affiliation, and we're having sort of a philosophical discussion now as to what this should actually mean, but as of now it's basically anything that we might determine uniquely about you or about the group of people you're in generates one of these affiliation tags, and that includes things like what courses you're in, what courses you teach, also things like what Unix groups you're in, which is really useful for controlling Web access and what sources you come from and so on and so forth. These just generate essentially individual strings, which are then all stuffed in under the affiliation tag.

Richard So to find everybody taking a sociology course, you would search the directory for this affiliation attribute containing that sociology course number or something like that?

Ben Yes.

Richard Okay. Does openLDAP support groups? Because that's another way that people are trying to address the same sort of thing.

Ben Not when we did the implementation. I don't remember offhand whether the newer version does, but we're a major version behind, so it's certainly possible that they have added it.

Richard Okay. And you said you have a Solaris directory where you look up incoming mail addresses initially?

Ben Well there are actually two things that get looked up. There's the mailing address that's attached to your directory entry, your general directory entry, that if you went to our LDAP servers as just some random person, if it was public you'd see, and so the mail system does use that, and also the other hidden field. So it uses that to determine if you have essentially a published mail address that we should be using. The other way that this comes in is we determine if you're a valid user on the system and determine whether or not we can write to your mail spool. That is just doing the standard Solaris `getpwnam()` call, but we are using LDAP as our name service switch package for that. It's a separately maintained LDAP directory though--they're two separate directories.

Richard So you don't feed into that from your other one every night or anything?

Ben Not really. They are tangentially related but conceptually they are essentially distinct.

Richard Okay. Any other questions or comments for Ben? Alright, well, thank you. Frank, you're last, so you're on deck now.

Frank Goals--generally a description of the project: Well, our directory project is quite old, it goes back ten years. And our initial project here at the University of Minnesota ten years ago was to give all faculty, staff, and students an email address. Back at that time access to the Internet was very spotty, it depended on what department you were in, the technology departments being very savvy and the humanities and whatever not even knowing what the word Internet meant. So the idea was back then to even the playing field and give everybody access to the Internet, give everybody an email address that they could use, and we went through that project. Part of it became obvious that if we were going to do that that you had to have a method to find out what somebody else's email address was without them having to--without having to call them on the phone and tell them. So the initial uses for our directory was first as white pages so that you could find someone else's email address, phone number, etc., etc., and also from a software standpoint we wanted to have simple email address in a form--like mine is `fjg@umn.edu`--to translate mail delivered to the actual system where I read and used email during the course of the day. So we wanted the sendmail software to be directory-driven and be able to convert these smaller more user-

friendly addresses into ones where it's pointing the mail towards the system that really was there. So those were basically the only goals we had day one, and we really didn't project ahead too much further than that because we were under the gun, we had three months from the time that we were told we needed to do this until the time that we were told that all had to be done in terms of providing email and directory services, white pages services for everybody. So we were pretty much under the gun and concentrating on solving that problem and nothing else.

The issue of--and it's been described in various way--perhaps political is the easiest way to describe it--of getting data from the legacy systems, was one of the more major issues involved. Our job at that was made easier by the fact that our CIO at the time was 100 percent behind the project because he was the one who wanted to give email to everybody, and so there was direction from top down that this was something that had to be done. But even with that top down direction, the data owners were legitimately concerned about how their data was going to be used and I think that anybody who's building a directory has to realize that what--they look to us techies sometimes as nothing more than a political stumbling block to really getting the job done--is in fact a legitimate concern on the part of the data owners as to how their data is going to be used, what's going to be done with it once it leaves their shop, etc. This is data that has to do with students and staff, they are responsible for it. So some of the notions of, "No, we're not going to be giving this data arbitrarily to just anybody who asks us for it." "No, we're not going to change it, any changes go to you to be changed and then you pass the changes on to us." Those types of rules and regulations with respect to how the data are being handled are very important to the data owners, and I think it's important for people who are putting together directories to know that those are legitimate and to respect them and to--and besides, in the end it, is to our advantage to say that we're going to--we're not going to change the data, the data has to be changed by the data owners, I think that's been said several times here as we've listened to other people, because we don't want to get in the business of having our own version of addresses and phones. We want there to be a database of record that has that accurate information, and it not be a situation where we get a feed from them once and then we're forever wondering whether the changes that we made to the data are better

than or worse than the changes that somebody else made to the data! It's got to be one person in charge of it, period, and that should be the data owners.

In our environment we get feeds for the entire University of Minnesota system, which includes all the campuses statewide. Initially our two feeds were from staff demographics and student registration. That expanded to alumni and to a few other smaller systems. Then there was also a contraction of that in that the staff and student feeds are now coming from the same database, which is PeopleSoft, because the legacy systems converted over the course of the last several years to a PeopleSoft system and thus from their side the staff and student data has been put together. So we no longer are really getting it from different sources, we're getting that from the same source.

The problems that--well, when we got it from two different sources for one thing, the ability of us to be able to identify that a person on the student feed and the person on the staff feed were in fact the same person or different people, our analysis of that basically came to the conclusion that we could not do so reliably, and so we created separate entries for staff and students back in those years. Now that we're getting the feed from the same source, both of them have the same unique PeopleSoft ID associated with them so that we're assured that we can know that they are the same person, and we are in fact right now in the process of going through and combining people who have two entries in our directory into one. They already have been combined in the sense that we have made use of the "see-also" attribute in the directory so that right now if there are two entries pointing to the same person they each have a see-also to the other one so that that's easily determine if someone's looking them up. But now we still get data from the alumni association, and there it can be hit and miss as to whether or not that we can determine that that person is already on a feed from another system or not. If we can determine it, we don't create duplicate entries. If we can't determine it, then we're forced into the situation of creating a separate entry.

On our side of things we do have a unique identifier that is completely ours to play with as opposed to the identifiers that we receive from the data sources. This allows us--I guess that way I look at it is that from

my point of view, my identifier never changes and everybody else's can. From their point of view of course their identifier doesn't change and mine does! So it's really a question of the point of view of the database. But it allows us to fix the issues where we originally think that it was two separate people and then realize that it's really one, or that we've combined two entries and said it's only one person and then later realize that it should be two.

What we use, however, the determination to get identify correlating identifiers and deciding whether it's one person or two people, that's done only once and that is when we initially receive a new person from a particular data feed. And I know that because that data feed always gives me what they consider to be their unique identifier. In the case of PeopleSoft, that's the employee ID, in the case of the alumni association--well I'm trying to think off the top of my head, I can't remember what they call it, but they have a unique ID as well, and so I look at that as being the unique identifier from them, and when I see it for the first time we decide whether we need to create a new person or should it be merged with an existing person on the database. Once that's done it stays that way until somebody goes in and manually goes, "No, no, our initial guesses were wrong and one person should be two or two people should be one."

That's really--I have to say, though, that the duplicate issues are not really one of our major problems around here. It usually pops up--the most common thing that pops up is that somebody has a second entry that they didn't know about and thus somebody started sending them email and never got the answer because they never used that email box. But the example I heard earlier where, for example, HR corrects or incorrectly changes a social security number, so the social security number changes from one feed to the other would not result in a new entry for us. We would always use PeopleSoft's unique ID as identifying the data that came from them and the social security number would be used to match up only on the first round of that data where we want to see whether or not we match it up with other people.

The collection of data that I have, I never knew what to call it. A few years ago I heard someone call it a repository and that sounded as good a name as any, but basically this repository of data that we have

and the collection of the data from all the various data sources is what we use to power the directory or feed the directory. We get changes from data sources on different schedules. Our primary schedule is the one from PeopleSoft, which is nightly. That means that hiring and quitting and firing and registering for classes and dropping out for classes, etc., is reflected in the next business day from when you did it. And so we get five feeds from PeopleSoft weekly. Some of the other data feeds can be less than that. The alumni association I don't believe gives us a daily--I can't even remember what the schedule for them is. There are other departmental feeds that we may get only when there are significant changes on the part of the department and they want to say, "Here's a new upload of data," where they're not really changing their data on a regular basis, they're sort of batch-feeding it so then they batch-feed us.

We do send batch feeds of data to other systems, like the library, food service, sports and rec, and things like that. All the feeds that go out from our site to them are ones that have been approved by the data owners. And in fact, most of those feeds are done from us as a convenience to both the destination of where they're going from and also from the data owners themselves, that they don't have to write interfaces to these other systems. The directory can be a much easier interface to go into groups like sports and rec and library and stuff than the original legacy system. This is especially true at the University of Minnesota during the time we were converting from mainframe systems to PeopleSoft. Having those feeds coming off the directory meant that during that conversion process, the legacy folks only had to worry about the conversion of feeds to us and they didn't have to worry about 20 different conversions to 20 different places, so that made life a lot easier for them during that period.

Software platforms: I am actually running an X500 directory from Syntegra. The protocols that we allow to interface with it, going back ten years ago, it was finger and ph, then of course gopher became popular and we had an interface for gopher. Now of course with the Web there's an http interface and LDAP is the current four letter word in everyone's lexicon, so we of course have an LDAP directory as well. I look at those protocols merely as the way to talk to the directory and the fact that it's an X500 engine in the background is

sort of irrelevant to the process because the applications don't have a clue, they're just talking protocol out there.

Let's see--legacy interactions, I think I covered that. Correlating identifiers, I think I sort of covered that a bit. The primary identifier that we use on our side, as we call it, the directory ID (or DID), and that becomes the person's distinct--part of the person's distinguished name, and with that never changing, it puts us in a position where future use of the distinguished name and a Public Key certificate for example is going to be practical because we know the distinguished name's not going to change for that individual unless there is some major cleanup operation needed on that person's data.

Pushing data back to legacy systems--we do push back the assigned user name or network ID or whatever else you want to call it, Unix user name, I don't know, whatever word you want to use to call it. That one piece of data is something that we generate upon receipt of data from them, so we push that back to them and that allows the legacy systems to know--if they want to send an email, they can just send it to username@umn.edu, they can build their application programs around that in PeopleSoft land and it makes it very easy for them. Other than that, once pieces of data, however, we don't send anything back to the legacy system at all. The email address is considered directory data and resides exclusively in the directory, it does not reside in any legacy system. By email address I mean the address where your mail is finally delivered, not the published, nice-looking one (fjg@umn.edu) And there are a few other pieces of data that we consider to be directory data, such as the URL of your homepage, your password is considered directory data, we certainly aren't going to be passing passwords back to PeopleSoft. Stuff like that. So there's only once piece of data we really ever pass back.

Multi-campus issues--well we've been multi-campus right from day one. And I would say that the acceptance of the other campuses using our directory services really became--it was by attraction. The same is true of departments around here is that our directory services had a reputation for reliability, for accuracy, for being useful, for doing things, and as that reputation grew the campuses certainly wanted to be a part of it and not go in their own direction.

The interacting with local and personal directories--we do have Active Directory here. My group actually is responsible for the top level umn.edu domain controller. We've sort of gone around trying different things with Active Directory to see what works best in our situation, and what we've really come up with is, that at the top level domain there are no people, but that the one level domain down from that, in other words, groups that want to join that top level domain, is where we populate people, and we do provide a service that a domain controller can specify the criteria that they want to determine that would force an entry to be created in their domain controller that matches up with one that is in X500. Then whenever the X500 entry changes in any way, that change is reflected down in their domain controller, and that includes new entries coming in that meets their criteria, and it also includes entries that no longer qualify for their criteria being flagged to be deleted. Notice I didn't say deleted, I said flagged to be deleted. And then they actually do the deletion when it's appropriate. We've learned from a sobering experience that you don't want to just arbitrarily delete based on data coming through because a good example is an employee who changes departments, that can actually go through HR to look as though someone quit and was rehired, and you don't want to just up and delete their entry one day. They tend to not like that, especially on a Monday morning.

Let's see, successes and failures, I would say--well one of the first things is never make any changes on Friday! They bite you every time. The thing that people hate the most is that if something for them breaks or whatever that they have the whole weekend before anybody can fix it for them, so we tend to make our changes to anything early in the week so the help line is on top of it. I guess I don't see anything that we've done that has been a horrendous failure. We've certainly made mistakes. My advice is that when you make mistakes you are public about it and admit it right up front and don't try to minimize it or hide it because if you do then people will just--people will catch on to that and then they'll no longer trust your service. So I would say the best thing to do with any failure that you have is be honest about it and move on and try not to have it happen again. We have been very successful with putting out the directory. What started out as just the white pages and email lookup has become the primary method for us for authentication and authorization here at Minnesota. Everything from modem pool, to library access, to getting into

PeopleSoft, to writing grant proposals, to moving money around in financial systems, etc., is largely based on your directory entry and the attributes that you have there along with application requirements on top of that. WebCT for example is directory-driven as well. So we've had a lot of success in using the directories to be the center of business activity here at Minnesota.

My advice to others would be to start slow and start small and if you don't have a directory yet, white pages is the obvious goal. Get a good white pages up, get that working. What will happen is when you get white pages up everybody will begin to complain about the fact that the data for them in the white pages is inaccurate and then they can learn how they fill out the forms to HR or student systems to get their data corrected. Then what'll happen next is HR and student systems will get way overloaded on processing forms to correct data and decide that, gee whiz, maybe they should put up some Web interfaces so people can do it themselves and they don't have to do it, and that will make the data even better again. These types of things I think grow on themselves. And so what starts out--your initial feed from places will not be the same quality of data that you get later. I can tell plenty of stories, everything from having I forget how many thousands of dead people I had in the directory the first time we put it up. The data like that will eventually fall through and get cleaned up simply because it can't stand the sight of day; that's the best way to get data accurate. Be careful, though, with that politically. I mean the other thing is you've got to always watch it from a political standpoint. It's true to say garbage in garbage out, but don't be caught pointing the finger at your HR too much or a lot of goodwill will go out the window. Be cooperative with HR, help them, don't make them look bad in order for you to look good. When people see something wrong in the directory, it's natural for them to blame the directory, and to some degree that's just fine, just say, "Oh, okay, fine," and then go back to--keep some of that interaction about data sources under the covers I would say and not--imagine yourself being back in HR when you start criticizing them for data! I think I talked myself out!

Richard Well, I think you covered everything pretty well. Thank you!
Keith, do you have any comments or questions?

Keith I guess I'm wondering, on the downstream systems that are getting feeds from you, maybe I missed it, do you send them the DID along with--

Frank No, the DID actually is--Other applications could use the DID if they wanted to for identifying who somebody is after authentication, but that's generally I would say most use some other identifier. I consider the DID to be sort of my thing. And the library, for example, food service, sports and rec, they don't really have any good, legitimate reason for a DID. Usually those groups have their own identifier that they want to see me feed. The library is--I can't even remember what they call it, but they have their own identifier and that's what they use to pick unique people out and they want me to send their feed with their identifier.

Keith And you're carrying that.

Frank And I'm carrying that, right. So basically from the standpoint of the directory, I have identifiers from all sorts of places and the DID sort of gets to be mine and when--because everybody needs a good placeholder. The way I describe my DID, of course, is that my DID never changes for somebody and PeopleSoft says, "Well, emplID never changes for anybody." And the library says, "Well the library ID never changes for anybody." But we each look at those other ones and say, "Well, you know, they do sort of change from our perspective."

Richard Frank, you mentioned a repository. What form is that? What piece of software?

Frank In our situation it's nothing but a bunch of index sequential files controlled by some cobol programs.

Richard Okay, so it's old stuff.

Frank Yeah. Nothing's very special about it at all, it is index sequential files. I can't imagine any need to get anything fancy like an Oracle database, it's almost really overkill for what we really need.

Richard And then how do you feel out of that into the X500 directory?

Frank Right now we're using secure LDAP to update Active Directory and we have on X500 we have--what's it called, I'm looking for the right word. Well, as luck would have it, I can't remember, it, but basically it's just a different protocol to update X500 out of it. Why don't we called it DAP, that'll look like I really know what I'm talking about! How's that! You won't tell on me will you Keith?

Keith My lips are sealed and we'll edit the transcript!

Frank Okay!

Richard And then you move data from the X500 directory into these Active Directories?

Frank Actually the point of move, if you will, is actually from the repository out to various places.

Richard Okay.

Frank If you look at--prior to having a name for it, I always just pretended like it was the directory, but basically there's data that is plenty of data, both objects, people, and data about people that are in the repository that are not in a given directory, whether that's X500 or Active Directory.

Richard Okay. Well good. Keith, anything else?

Keith I really think that's probably a wrap and it's really good stuff.

Richard Yeah, I think we should wrap up then. Ellen, are you still there?

Lisa Ellen's gone, but Lisa's still here.

Richard Okay, well, thanks for your help and I hope the recording comes out okay today.

Lisa Me too.

Richard And Frank, thank you.

Frank You're welcome.

Richard Ben, are you still here?

Ben Yeah, I'm still here.

Richard Great, well thanks for participating.

Keith And we look forward to you comments on the transcript draft and Richard's going to be working with the rest of us but mainly him on some good practices if not best practices here, so we'd appreciate your chiming in on that as they come out. So we'll make sure that you guys get copies of those, right Richard?

Richard Right.

Keith Yeah. And thanks, it's really great. And we're all in the same big mud puddle!

Frank We're all squealing happy right!

Keith That's right. If you like mud, we've got mud!

Frank Yep!

Richard Okay, good bye.

All Bye.