;login:

THE MAGAZINE OF USENIX & SAGE
April 2002 • Volume 27 • Number 2

inside:

SYSADMIN
Wiki, Blog, Zope, and Other Communicative Grunts
BY STRATA R. CHALUP

USENIX & SAGE
The Advanced Computing Systems Association &
The System Administrators Guild

wiki, blog, zope, and other communicative grunts

A Quick Look at Some Modern Collaboration Tools

by Strata R. Chalup

President, VirtualNet; Starting as a Unisys 68K admin in 1983, Strata Chalup is now an IT project manager but allegedly has retained human qualities. Her mixed home network (Linux, Solaris, Windows) provides endless opportunties to stay current with hands-on tech.

strata@virtual.net

I've been using some interesting Web-based collaboration tools for about 18 months now, and have been meaning to write about them. Hal Pomeranz gave me a good excuse when he obliquely mentioned Wiki as a tool at LISA last December.[1] I heard several folks afterward saying "Wiki? What's that?" and thought to myself, "That's right, I never did finish writing that up!"

A few of these tools require you to use JavaScript on your client browser, and I will warn you when that is the case. Most, however, are fully Lynx compatible and can even be used with Netcat or Telnet to port 80 if you are really in a pinch. I will say that doing Web-based collaboration by cut-n-paste over a Telnet connection is not necessarily in the spirit of easy update and publishing that drives these tools. We're an odd bunch of ducks, though, and it's nice to know that if you absolutely have to do something like that, in many cases you can!

Patterning Toward Collaboration

"Wikiwiki" is colloquial Hawaiian for "quick." The full name of the first wiki was the WikiWikiWeb. Wikis were invented, largely in collaboration, by the object-oriented programming community. Dale Cunningham, a founder of the Portland Pattern Repository, has been credited with much of the design and implementation. I believe at this point that many of you are familiar with patterning as a design and/or soft-skill tool in programming. A capsule summary is that in the early '90s, many of the SmallTalk luminaries realized that they could apply to language design (and later, to software engineering) the concept of "patterns" captured by Christopher Alexander and the Berkeley Design Group in their seminal work *A Pattern Language*. While Alexander's work was strictly in physical architecture, i.e., buildings and landscape, the idea of meta-patterns was so intuitively useful that it has become an integral part of object-oriented language design and software reusability engineering. The Portland Pattern Repository was founded as a collaboration space for language and software pattern development.

In system administration, patterns would be the things that come out of an applied taxonomy, such as best practices or common failure modes. As work on system administration taxonomy continues, primarily pioneered by Geoff Halprin and Rob Kolstad, sysadmins will begin to join the patterning community and develop pattern repositories of our own. I gave a short talk at BayLISA in 1998 about sysadmin patterns and the pattern concept in general. My interest in wiki and similar collaboration tools springs from my desire to see the sysadmin community start to create our own patterns. It is my hope that the recent progress on taxonomy and growing a "Book of Knowledge" for our profession will get us to the stage where we can start explicitly patterning in a collaborative process during 2002. Some of the modifications I'm working on for AtisWiki are specifically oriented toward enabling this process, but I'm getting ahead of myself here. Let's define a wiki first, and then start ripping into it.

NOTE

[1] During his acceptance speech for the SAGE Lifetime Achievement award – yay, Hal! Congrats! :)

The Wild and Wonky Wiki

So, what is a wiki? It's easy to confuse a wiki with many other things which it is not, so let's build up the definition in stages. In the most basic description, a wiki is an interactive Web-based authoring system. It is most often implemented as a series of CGI scripts acting on a shared document repository. The repository usually consists of flat ASCII files, which may or may not contain HTML or HTML-like tags specific to a particular wiki implementation. Often there is a source code control system behind it used for check-in and check-out by the scripts. On more advanced wiki environments, there is the concept of user authentication, automated tagging of updates (date, time, user), per-user locking of files, access control, and various degrees of granularity of read-modify-publish for individual documents or subtrees.

A wiki is an interactive Web-based authoring system.

This is indeed a somewhat pedestrian description, and if that were all there were to a wiki, we'd be bored and stop right here. Now for the next layer, which is indeed more interesting. What makes a wiki so fast and easy to use, that is, why is it "wikiwiki" (quick)? When a document is saved or displayed, the parsing rules are used as a shorthand to determine if something is a hypertext link or not. The parsing rules are generally configurable within any wiki implementation, but the most common rule is that a mixed-case word is a hyperlink. Thus "ServerConfiguration" would appear as a link, and "server configuration" or "serverconfiguration" would not.

Why is this useful? Because if there wasn't already a ServerConfiguration page, the wiki would now create one for you. The link to this blank page would be immediately distinguishable. Unlike a regular link, which is simply highlighted in traditional Web fashion, an uninitialized "blank page" link is modified with a configurable pattern, usually "_?" added to the link name. Thus our blank page would show up as "ServerConfiguration_?" when the saved document was redisplayed. There are also shorthands for common types of markup, like beginning a line with "*" or "**" to create a bulleted list. Types of markup supported vary by wiki implementation, of course.

Although the authoring system is Web-based, the documents are not necessarily HTML documents. Most wikis have options to include HTML code (and display it properly) on pages after editing them. Some will include several different markup systems, including ASCII text, and let you choose to which markup group the page is assigned. Choosing the wrong type for what is already on the page can result in a page that will not display. When I was running a 9/11 information site wiki this September, I had folks trying to edit the pages and changing the type to "plaintext," which nuked all the links, or to "transclude," which nuked ALL the HTML and left the page uneditable until one constructed an "Edit" URL by hand. The "Keep It Simple, Silly" principle is a good one and probably dictates that wikis won't attempt to sanity check or force page markup types anytime soon. In the meantime, experimenting in the wiki "SandBox" area and setting the page wrong can ruin your whole day, especially if you're a newbie. I consider that a serious conceptual flaw. Which brings us to . . .

Wiki Warts

A Maze of Twisty Little Passages, All Different

As you can readily see, the ability to grow a document repository in a very organic, ad hoc fashion can easily turn into a double-edged sword. I'm an AwldPhart, and I like to see some kind of master index or master map to EveryDarnPageinThisWiki; unfortunately, most wikis do not support such a thing. The best one can do is to go back, often way, way, waaaayyy back, in the RecentChanges page. As its name implies, it is updated when any document in the repository is . In most wikis, it is created automatically, but

The WikiWay, as practiced in the patterning and OO communities using wiki, is anathema to control freaks.

not rotated automatically, so it is either very very long or has been hand-maintained and moved into month and year pages. Obviously, then, RecentChanges is also a valid wiki page, meaning that anyone can go in and screw it up, excuse me, edit it by hand.

A modification to some wikis is the keeping of a parallel index, essentially a content index as well as a RecentChanges. I'd like to see mods that kept a MasterIndex by date, as well as a user-tweakable ContentIndex. There's no reason why a wiki can't update multiple pages on a save command, although the more "shared" or "master" pages, the more likely one is to run into locking or halting issues. I think that as time goes on, features of threaded mailing lists will find their way into wikis, such as having all pages by a particular author linked into an Author page. This is lagging, as we will see later, because the concept of registered users and authentication is not widely practiced in the wiki world.

Who Moved My Cheese?

The ability (and tacit permission) of others to edit one's content, rearrange it, and generally move it around can induce a white-hot stellar phoenix reaction in those of us accustomed to having info stay where we put it. The WikiWay, as practiced in the patterning and OO communities using wiki, is anathema to control freaks. Can all two or three of the sysadmins out there who are not control freaks please raise your hands? Exactly.

What's the fix for this? None, on a public wiki belonging to another community! Content yourself with an RCS- or CVS-capable wiki, of which there are several, and know you can pull your info out of the Backups section of the document page. The other way, which irritates the "philosophically correct" wikians, is to use some combination of directory permissions and modified wiki scripts to enforce an author/commentator split between content updaters.

With Stone Knives and Bear Skins . . .

Another aspect of wiki use which will put some folks off is the lack of a real editing environment. The wiki Edit page in most implementations is simply a free-form text box which is pre-loaded with the existing page content. If you are doing markup in HTML, you're editing the tags by hand. This is not necessarily a problem for most of us, myself included, who still do a lot of HTML in vi. You will find that it grows very tedious when updating a large page. This is especially true when your text is pasted in from elsewhere and you don't realize it contains things that could be taken as WikiFormat: spaces and asterisks in a traceroute output, for instance, parsing as nested <UL> type lists.

Some wikis (such as TWiki) include a Preview button so you can see if your edits look reasonable before you commit. Actually, those which include a Preview button sometimes make that the only button, forcing you to do a preview step before being able to Save. That gets annoying very very quickly.

Where There's a Will, There's a Wiki

One thing which has kept many of us from joining the "Web revolution" with any enthusiasm is a mind-set which embraces the command line, two-way ASCII communication, and keeping things scriptable and simple. Of all of the tools we're going to mention today, the wiki is the most UNIX-philosophy friendly and easily tweakable in your scripting language of choice. While there are many wiki implementations in "newfangled" immersive environments like Squeak or Zope, the majority out there are Perl, Tcl, and Python. There are even a few done in C, suitable for revamping and customization by the most stubborn (or overcommitted) fossils who haven't picked up much Perl yet. Who, me?

My preferred wiki at this point is AtisWiki, a fairly simple variant implemented in Perl as CGI scripts. Wiki variants are well suited to becoming Apache modules. I have found AtisWiki is fairly efficient when run under the mod_perl facility in Apache. I am not alone – there is now a UserModWiki, currently at version 0.92, which started with AtisWiki and made it an Apache module. Note that AtisWiki uses RCS natively, yet file locking is not strongly implemented – the HTTPd user is checking out the files, and there is currently no concept of sessionization that I can see in AtisWiki. I'm hoping that the UserModWiki developers have this on their list of features!

I still prefer AtisWiki for my own applications, though, because it is so straightforward and easy to customize. I am planning to integrate an alternate-channel publishing step into the AtisWiki "save" process. Instead of only updating the viewable Web page, one could also trigger additional actions upon a successful save. These actions would be based upon the (valid HTML) meta-tags in the document. The integration of existing Perl code for an RSS feed would be a tremendously handy application for a wiki, especially given the traditionally close coupling between the Webstream publishing communities and the wiki communities. In fact, it's something of a mystery to me why this does not already exist.

RSS is a wonderful broadcast medium for things like system error messages or logging info and alerts.

RSS is Rich Site Survey protocol, essentially a "headline news" format which supports keywords and meta info. Developed originally by Netscape in ancient times (i.e., the mid-'90s), it largely languished, but its potential is vast. A minor squabble over the future of RSS, and a brief interlude of incompatible protocols, has settled out nicely. RSS is a wonderful broadcast medium for things like system error messages or logging info and alerts, although it is currently used almost exclusively in a Web publishing context right now. Check out Jeff Barr's Headline Viewer tool, or the Syndic8.com site for aggregating and rating RSS feeds. Try using Headline Viewer or another RSS infeed viewer to get your SlashDot, FreshMeat, and similar news. You may never go back to those giant page views!

Another variant which has great appeal for me is DolphinWiki, implemented in C. It's currently a "closed" source wiki, which in this case means you have to ask the developer for a copy. He's a nice guy, has written a nice wiki, and will send you a copy if you ask nicely! DolphinWiki has some features I especially like, most notably closer tracking of document changes than AtisWiki, and some of the navigation/auto-index features I mentioned in my wish list above.

BayLISA is currently using a wiki variant called TWiki, which comes with some interesting presets for an office or work-group collaboration. We keep our speaker prospects and schedules in the wiki, and have been adding "legacy knowledge" culled from mailing list archives and other missives that have had no clear home. The ad hoc nature of the wiki makes it easy for one of us to spend an hour or so on a slow afternoon putting up a document area with useful bits that previously had no real "home." It's a lot more intuitive to make a page and add to it by creating other pages, like MeetingOpenChecklist, than it is to create subdirectories in a file tree and try cute naming games with filenames to try to convey the same information. Since we authenticate to log into the TWiki, it tags the files with the creator, so we know who to ask if something is not clear.

Of course, since TWiki (and all the other wiki implementations I have looked at) has no concept of static Web pages, the wiki info and the baylisa.org pages routinely get out of sync and must be fixed by hand. This is particularly irksome since we have a

Zope is, depending on whom you talk to, a toolkit, a language, a philosophy, a way of life, all things to all people.

makefile and script that update the speakers list, the library files, and similar info. As I mentioned above, when we finally combine wiki interactive document creation with a push/publish model à la Blogger (see below), we will *really* have the next generation of collaboration services.

You may be interested to know that there are more co-development projects associated with sysadmin and IT for TWiki than for any of the other wiki implementations I have examined. One which may be of particular interest to *;login:* readers is the TicketWiki, a trouble-ticket system implemented using TWiki. One can create normal wiki pages, or new tickets, and there are some goodies for tracking tickets and modifying them, and enough customization hooks to let you put on a fuzzy suit and stick yourself onto the wall like Velcro.

But Wait, There's More

Blogging: Blogger, by Pyra

Blogger is a Web publishing service which lets you create a page template and then add arbitrary text and markup to it via a publishing bookmark in your browser. The bookmark itself is a script which takes whatever you've selected on a browser window, plus the URL of the window itself, and uses that to fill in the template, then plunks you in an edit form via the site. You've undoubtedly seen many Web logs in the news since 9/11, so are familiar with the format. I keep several (neglected) public blogs, and also some private blogs where I keep notes to myself. When I'm scanning through a lot of Google or AlltheWeb searches to learn about some technical problem or tool, Blogger lets me pull the important excerpts from a page and capture them, along with the URL and my commentary on what I'm seeing. It's a great way to create annotated catalogs of resources as well.

Easy to use, easy to publish, easy easy easy. I love Blogger, though I don't use it as much as I'd like these days. Why? You have to use JavaScript, for one, and it updates via FTP for another. Wups. The smarts don't reside on a server you control yourself, but that's not a fatal flaw for my application. On the bright side, the content lives where you want it, on their site or on yours, and the service is free. Even if blogger.com were to go poof, you'd at least still have all your pages, and might be able to whip up some kind of replica on your own.

Immediacy is the core of blogging, yet an email-based blog would be very handy indeed. How would it be different from a standard mailing list archive? The shiny candy-coating formatting around the goodies, plus whatever you want to build into it. Maybe someone will build this as a gateway service. The first blogger add-on of which I'm aware debuted recently, namely "BlogSkins,", a forum for swapping blog formatting templates. Go team!

Zope! The Python and Squeak people . . .

The folks who love Zope are off on FreshMeat building all kinds of way cool things. Zope is, depending on whom you talk to, a toolkit, a language, a philosophy, a way of life, all things to all people. Okay, it's not that bad yet, but there's a lot here to dig into. If I had to do a one line summary, I'd say it's a Web/markup language in the same way that Perl is a string manipulation language, but that doesn't do it justice.

I'm not much of a coder these days, and tend to be rusty at learning new languages, so I have little personal experience with Zope beyond downloading a few toolkits and saying, "Whoa, cool!" Oh yes, and listening to my Web publishing friends rave about it.

If you are into Python, or remember Common Lisp with nostalgia, or are bursting with the joy of Squeak, then Zope is the next new toy for you.

The Zope user community can't be beat – they are warm, welcoming, and reach out to newbies in a variety of formats. Everything from Web logs to ZDP (the Zope Documentation Project) is out there to help you get up to speed.

Scripting Radio: Radio Userland, by Dave Winer and friends

Radio Userland is an immersive scripting and publishing environment powered by Manila, the same engine used for Dave Winer's commercial authoring environment called Frontier. I have found it rather impenetrable so far, but many programming-oriented friends have enthusiastically embraced it. Available only in Windows and Mac for most of its evolution, Radio Userland has features that include native use of XML, extensible scripting, built-in push authoring and blog-equivalent tools, private file-sharing, and more.

Drawbacks include a support community (especially the author) that seems to be inclined to be a bit defensive, and the current lack of any UNIX versions. Linux is supposedly "in the works," and it's a good idea to check for updates frequently. I think that checking out Radio will inspire folks to make Zope be all that it can be, but some of you may say, "Hey, Radio is a better fit for me." Dave, don't hate me! :-)

No More Mailing List Drift! QuickTopic, by Steve Yost

This is not an immersive publishing tool like the others, but it is a handy ad hoc publishing environment that complements email and is such a neat idea that I mention it here. Formerly known as "Take It Offline," Steve's system allows you to authenticate, start a topic, and then notify the mailing list or individuals that the topic exists. You can read the topics on the Web, or be notified by email of new postings. The emails from QuickTopic have routing headers constructed on the fly to funnel replies back to the topic.

Right now the service is free, and topics persist somewhat indefinitely, though ancient topics may someday be cleared to create more room. QT is extremely handy in keeping a mailing list focused while allowing folks to participate in specific off-topic threads with the rest of the mailing list community – at least, the ones who are interested! The system tracks all of your subscribed topics for you so that you can aggregate them onto a single page for browsing, although most folks get updates by email.

Why isn't this just Yahoo Groups? Because the focus and design is on a temporary home for ephemeral information, in conjunction with a mailing list community. Plus being completely free, voluntarily user-supported, and (so far) without advertising. QuickTopic is very lightweight and is not trying to be the kitchen sink. Your mileage may vary, but I find this handy enough to mention.

And was there a point to all this?

You bet! While most of us are on the cutting edge of technology in our own fields and specialties, the way we sysadmins communicate, as a profession and amongst ourselves, tends to be more in the form of jungle drums or carrier pigeon! There are a lot of amazing tools out there that we could be using in our community, and I hope that I've brought a few of them to your attention. Let's go out there and have a little fun.

BIBLIOGRAPHY

AtisWiki:
http://www.ira.uka.de/~marcus/AtisWiki.html

Blogger, by Pyra:
http://www.blogger.com/

BlogSkins:
http://www.blogskins.com/

Dolphin, a C wiki:
http://www.object-arts.com/wiki/html/Dolphin/FrontPage.htm

QuickTopic:
http://www.quicktopic.com/

Radio Userland:
http://radio.userland.com/

Radio Userland – Dave Winer on publishing:
http://www.scripting.net/

RSS – Jeff Barr, RSS Godling (hi, Jeff!):
http://jeffbarr.editthispage.com/

RSS – O'Reilly's RSS info site:
http://www.oreillynet.com/pub/a/rss/

RSS project – NewsClipper:
http://newsclipper.sourceforge.net/

RSS uses discussion:
http://discuss.userland.com/msgReader$10115

The Wiki Way: Quick Collaboration on the Web, Ward Cunningham & Bo Leuf, Addison Wesley.

TicketWiki:
http://TWiki.org/cgi-bin/view/Codev/TicketWiki

TWiki main page:
http://TWiki.org/

UseModWiki: *http://www.usemod.com/cgi-bin/wiki.pl?UseModWiki*

WikiWeb entry point:
http://c2.com/cgi/wiki

Zope Development Home:
http://dev.zope.org/

Zope Documentation Project:
http://zdp.zope.org/

Zope newbies site:
http://www.zopenewbies.net/

ZopeLabs (Zope cookbooks):
http://www.zopelabs.com/