Doco

Description

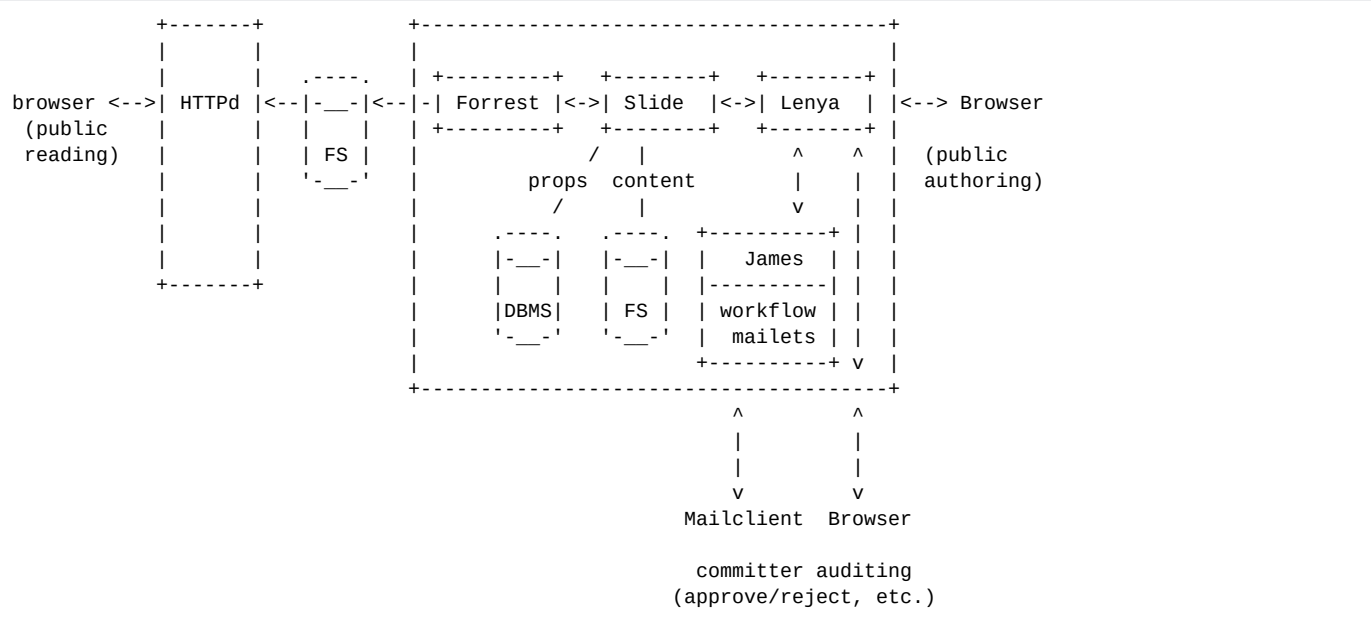
The proposal is about the creation of a content management system for apache projects, codenamed "Doco", whereas the priorities should be

- Ease of use
- Security

Features at a glance

1. super-easy editing (should satisfy wiki lovers)
2. minimal, efficient and secure workflow (should satisfy board@ legal concerns)
3. should allow the creation of static content (should satisfy infrastructure@ load concerns and mirror@ concerns)
4. should aim to reduce the production of custom code to a minimum (all code should go back to the projects that it depends on, if makes sense to do so)
5. should come up with structured XML content well organized in a versioned repository (should aim to make content long-lasting and easily repurposed)
6. should aim at complete internationalization of the content

Architecture at a glance



Details

Editing

1.Every public page will have a "edit" link.
1.This link will forward to a page that will include a WYSIWYG XHTML editor built inside the browser (Linotype, currently works both on IE5+, Moz1.3+, Firebird 0.6+, no software needs to be installed).
1.The editor will have a login form at the bottom:
1.#if you are a committer, the changes go straight into the repository
1.#if not the workflow is started (there will be a hard-to-guess text image that the user has to type in in order to avoid spamming bots)

Mail Workflow (thru James and custom mailets)

1. If document is submitted by a non-committer, email is sent to the 'documentation committer' list (which is a private list) for that particular project
2. Committer will receive an email similar to moderate ezml moderate emails and diff between published and authoring/working document are being shown. , e.g.
   *project-subproject-accept-1067323167.74085.bhhdiekdcmdijjeaiknc@doco.apache.org
   *project-subproject-reject-1067323167.74085.bhhdiekdcmdijjeaiknc@doco.apache.org
3. if three "accept" and no "reject" emails are obtained, the diff is allowed into the repository
4. if at least one "reject" is found, the mail workflow is stopped and workflow has to be resurrected by connecting to the backend with a browser

Web Workflow (thru Lenya)

1. thru a web interface, a documentation committer for a particular project has the ability to undo changes and resort to previous versions of the documents.

Content Repository

1. must have WebDAV interface (to allow language abstraction and installation on different machines)
2. must have DeltaV linear versioning or autoversioning (to allow rollback in case bogus or vandalized content sneaks in)
3. must have DASL capabilities with ability to run queries against live properties (which will be used by the workflow system)

Slide is the only open source project that is able to support all these things. (Note: subversion does 1# and 2# but not 3#!!)

Publishing System

1. must generate static documentation
2. must be able to operate on a WebDAV repository thru a DASL layer

Cocoon gives DASL quering capabilities that Forrest can use to obtain the XML documents from.

Issues

Editing

1. how do we edit navigation structures (sidebars)?

How about using the hierarchical repository structure for that?? since the repository structure has to be managed somehow anyway... DavidNuescheler.

A hierarchical structure indicates a unique placing, or, given the ability to create symbolic links, a preferred location for that resource. This also implies that someone (might not be the author of the page, but still a person) must decide where the page resides. This decision is often the most expensive in decisional terms, because its hard to measure the judgment made, especially in environments like open source where there is no figure for an editor. Also, the location of a particular resource in a hierarchy seems to limit the ability to reuse that resource in another context and, potentially, without human intervention... Stefanomazzocchi

Hmmm... I would argue that hardlinks in a hierarchical repository structure would give you the ability to display as many views on top of a hierarchy as you like, and by no means imply "human" intervention. In my mind the hierarchic unix-filesystem metaphore (with sym and harlinks) has not only proven to be one of the most successful and best understood models, but is also extensible beyond the obvious, thinking of things like /dev or /proc DavidNuescheler.

In Daisy, a navigation structure will be edited as a 'normal' document (i.e. XHTML) containing a (possibly nested set of) unordered lists. That way, people will have the ease of editing the navigation tree using a normal editor. – StevenNoels

TopicMaps]: [http://www.topicmaps.org/xtm/1.0/#conceptualmodel are designed precisely for this. Forrest has a similar idea with a separate map of topics.

The source documents for the topic map could be a variety of things, XHTML docs containing lists, UML class diagrams drawn in dia, harvested metadata, or anything, and transformed into XTM (the standard XML syntax for Topic Maps). These could then be merged to create a complete map.

From the ISO/IEC 13250 Topic Maps spec (a huge PDF, so I'll quote the entire section on "scope", rather verbosely):

_Topic maps enable multiple, concurrent views of sets of information objects. The structural nature of these views is unconstrained; they may reflect an object oriented approach, or they may be relational, hierarchical, ordered, unordered, or any combination of the foregoing. Moreover, an unlimited number of topic maps may be overlaid on a given set of information resources. Topic maps can be used:

- To qualify the content and/or data contained in information objects as topics to enable navigational tools such as indexes, cross-references, citation systems, or glossaries.
- To link topics together in such a way as to enable navigation between them. This capability can be used for virtual document assembly, and for creating thesaurus-like interfaces to corpora, knowledge bases, etc.
- To filter an information set to create views adapted to specific users or purposes. For example, such filtering can aid in the management of multilingual documents, management of access modes depending on security criteria, delivery of partial views depending on user profiles and/or knowledge domains, etc.
- To structure unstructured information objects, or to facilitate the creation of topic-oriented user interfaces that provide the effect of merging unstructured information bases with structured ones. The overlay mechanism of topic maps can be considered as a kind of external markup mechanism, in the sense that an arbitrary structure is imposed on the information without altering its original form._

There are some Cocoon components for dealing with TopicMaps already (part of the tm4web project at SourceForge).

Staging to Live

1. how do we transfer the content to the main web machine? croned inverted rsynch? direct transparent proxying/caching?

under the assumption that i understand the issue correctly... would the "reverse proxy" remove forrest from the picture?? reverse-proxying on the live machine certainly would make the roundtrip from the author to the live site much more responsive which should increase sexyness? do you think the apache infrastructure group will like the dependency to a dynamic (java based) system? if yes i think it might be beneficial to introduce a push based cache invalidation for mod_cache... DavidNuescheler.

the mod_proxy/mod_cache couple are definately the way to go in a dynamic environment, still, from both a political and community perspective, I think that using static content is the way to go for now. But I do agree with you that in the future a transparent mod_proxy/mod_cache situation with push-based (and IP-filtered to avoid DoS!) cache invalidation mechanism for mod_cache would be extremely beneficial, for Doco and for many other situations, but at a later stage...Stefanomazzocchi

Workflow Issues

1. how can we make the channels more secure? SSL on the editing part and PGP/SMIME signing the mail replies?