CocoonDocumentationSystem

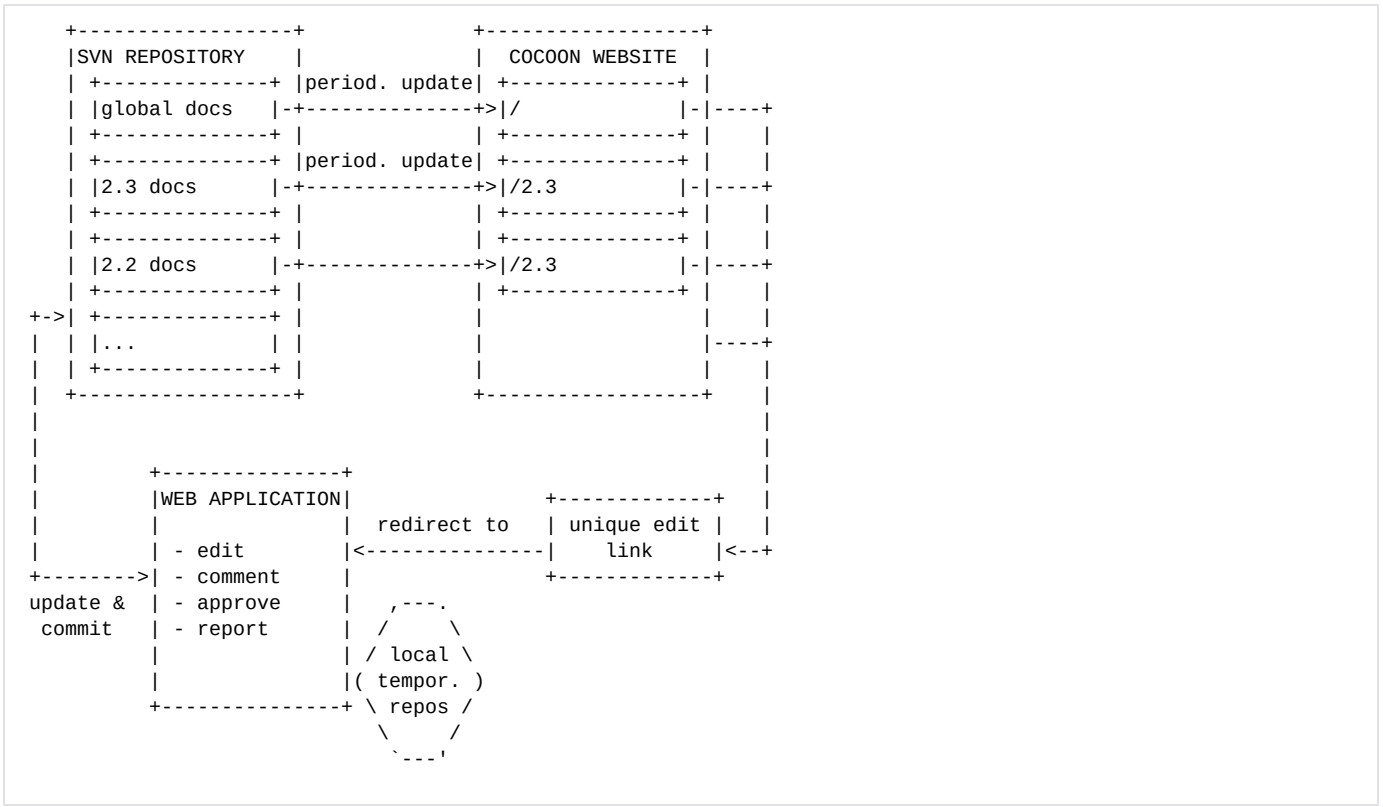
Motivation (or problems I want to solve)

- make writing docs as simple as possible
- separation between global docs (docs about the project) and user docs
- evaluate the existing docs (no automatic doc import into new repositories)
- enable comments for doc
- have a modular architecture (having a repository and webspace should be enough to run, other features should be "pluggable")

I don't think that a technical solution will directly lead to better docs. I just think that making the doc authoring process easier will invite much more people to help out.

Architecture

Overview



The center of the new architecture is SVN. SVN contains all Forrest repositores. For the first, these are[1]:

- 2.1 docs
- 2.2 docs
- global docs

There are two ways how the docs of these repositores are published:

- periodicially using some kind of automatism (cron, forrestbot)

Let's look at the structure of the website:

```
http://cocoon.apache.org ................... contains the global docs
http://cocoon.apache.org/2.2/............... contains the periodically (e.g. every 6 hours)
                                             published docs
```

Note that publishing multiple versions is good but can lead to confusion when people search on the web and get a page from a different release than what they are using. I'd suggest a note/link on each page, clearly stating to which version of Cocoon this page applies, and giving access to the version navigation (ideally a link which searches for the same info in other versions, but that's for a future release 😉 - BD

After the discussions on dev@cocoon I dropped the idea of publishing patch release docs. See the new drawing above that already reflects this. If you're interested in the original version of it, look into one of the former versions of this wiki page. ReinhardPoetz

---

The "living docs" docs that are periodically published out of the current repositories contain an "edit" and a "comment" link. This link is redirected to a web application where authors can write new docs or comments, e.g.

```
http://cocoon.apache.org/edit/2.2/23.html --> http://someapacheserver.apache.org/webedit/edit/cocoon/2.2/23.html

http://cocoon.apache.org/comment/2.2/23.html --> http://someapacheserver.apache.org/webedit/comment/cocoon/2.2
/23.html
```

The web application at http://someapacheserver.apache.org/webedit/ has following features:

- edit a document e.g. document 2.2/23.html (everybody)
- save the modified document in a local repository (everybody)
- commit the document to SVN (committers only)
- get a list of all locally available versions and the latest SVN version of a document so that the author can select from which version he wants to start (everybody)
- add a comment to a document (everybody)
- commit the comment to SVN (committers only)
- send mails to docs@cocoon.apache.org when a new doc or comment has been written
- make site.xml editable using a nested list of <ul> (committers only)
- user management (committers only)
- report: list of all modified/new docs and comments available

The implementation of this mini CMS could be done based on

- Daisy
- Hippo CMS (if it is OS licensed in the meantime)
- Apache Lenya
- Forrest
- Gianugo's little CMS
- write my own

_Look at http://wiki.apache.org/cocoon/CocoonDocumentationSystem and find all my requirements. I'm sure that all six options are good enough but as *I* have to do it, I'll take the road that's the fastest for *me*. Don't forget that learning where I have to add extensions to an enterprise level CMS like Daisy, Hippo or Lenya would take *a lot* of time. And as I have the (maybe illusionary) view, that I would be rather fast in implementing it on my own, I'll probably go the Forrest, Gianugo or "write my own" way. That's it.

Once again, nothing speaks against taking one of the enterprise level CMS and if one of the communities around them does the implementation and takes all requirements listed at http://wiki.apache.org/cocoon/CocoonDocumentationSystem into consideration, I'm the last one who stops them. One thing to add: I want to see a prototyp of the webapp running in 6-8 weeks from now; whoever writes it._ ReinhardPoetz (a reply to a mail of Andreas Kuckartz who asked what would speak against Apache Lenya)

---

[1] If 2.2 is coming soon we should concentrate on 2.2 docs

Document format

I propose the format that the default configuration of the HTMLCleaningConvertor generates. What is good enough for Daisy should work as well for us 😉 Alternativly we can use plain HTML4. This would have the advantage that the document can be edited using any HTML editor.

I propose HTML as a base format. Forrest is able to render it without problems, and the Incubator website uses it as a source format. - NKB

Currently I think of supporting both, HTML (17.html) and "default HTMLCleaningConvertor XML" (17.html). Only one can be available, which is checked by Forrest. I think that after the online scenario is running, everybody will use it. Also if you like to write a first version with Mozilla Compose because you can copy the text into the HTMLArea form field afterwards. But for the time until the online scenario is running, we need an alternative, and here HTML comes in. ReinhardPoetz

---

In the repository, each document contains of 3 files:

```
17.xml ..................... contains the content
17.meta.xml ................ contains meta information (date, authors, history)
17.comments.xml ............ contains user comments
```

Splitting up the docs into three parts has the advantage that the structure of each document is simpler, editors can be used to edit the content only. Also Openoffice goes this way.

The SimpleContentModel idea might also be good, it uses a slightly different but interesting model, where a documents is either a single xml file or a directory containing the main document and additional files - BD

This is what I propose, that does not contain a metadata file. The author and dates infos are in the source file, while the history is in SVN... no need to replicate stuff. The comments stuff can be added as a Forrest plugin. Maybe putting the comments in a doc.comments directory, with each a separate html file would be nice. - NKB

```
17.html ....................... contains the content
17.comments.xml ..............  contains user comments
```

For the comments stuff, see my proposal in the Cocoon whiteboard. I simply added it in a custom pipeline - see http://svn.apache.org/repos/asf/cocoon/whiteboard/doc-repos/2.2/src/documentation/sitemap.xmap. About skipping the meta infos, I'm not sure. I also want to provide info about status, target audience, keywords and this for all languages. Putting this information into plain text makes it hard/inpossible to explicitly use it in the publishing process query it in the future. Here the structure of a document on the filesystem:

In the repository, each document consists of a directory, which can contain following files:

```
 ./content_en.xml ....................... content as cleaned XML
 ./content_en.html ...................... the content as HTML (only content_en.xml OR content_en.html are
allowed)
 ./content_de.xml ....................... content in German
 ./meta.xml ............................. contains meta information (date, authors, status, target audience,
keywords)
 ./comments_en.xml ...................... contains user comments in English
 ./comments_de.xml ...................... contains the user ccomments on the German version
 ./mimes/ ............................... contains all images and attachements
```

The changes include the ideas from Bertrand and Nicola. Additionaly I added a location for mimes (images, attachments like ZIPs) and the option for having the content in multiple languages available. ReinhardPoetz

Why 'mimes'?? 'attachments' or 'files' sounds less surprising to me - BD

Document identifiers

I think we should move away from speaking names like "custom_components" and use plain numbers and put all documents into a flat directory. Speaking identifiers are nearly always a problem in data modelling.

Advantages:

- no misleading document names
- several navigations can be provided
- during the life of a document, a speaking identifier may become out of date

Note to auto-generated docs: Every process that generates docs automatically, has to use a unique numbering scheme (namespace) so that IDs can't conflict with existing docs.

Apart from the very controversial idea of having numbers as IDs, the most important part is the flat structure (WIKI style) of all documents. (see http://marc.theaimsgroup.com/?l=xml-cocoon-dev&m=110593998230556&w=2 by Stefano). Structuring of content is IMO not a concern of the repository but of the publishing process. Forrest offers everything we need. ReinhardPoetz

After reading other's opinions I withdraw my proposal, and will use flat URLs with speaking identifiers. ReinhardPoetz

---

Forrest repositories

See http://svn.apache.org/repos/asf/cocoon/whiteboard/doc-repos/ for two examples that work with the latest SVN version of Forrest 0.6.

Published docs

The proposal for new global docs and user docs are available at http://apache.org/~reinhard/cocoon/1.html. Note that the page is generated out of two forrest repositories.

The global repository is responsible for the tabs "Projekt" and "Community". The tabs "Getting started" and "Documentation" link to the most recent, editable userdocs. Older (frozen) docs get their links in the second-level pelt in the "Documentation" tab.

What do we have to do?

Step by Step

The good news is that we don't need all the features at once. We can go the path to better docs and a better documentation system step by step:

- Step1: Setup Forrest Repositories
  - work on the new docs
  - evaluate good docs and move them into the new repository
  - find a new structure
  - integrate auto-generated docs
- Step2: enable the periodically running jobs that publish repositories
- Step3: work on the web application

Who does the work?

- work on the new docs (writing new ones, update existing ones) (Upayavira)
- setup the new Forrest repositories (ReinhardPoetz)
- initial **proposal** for a new structure (ReinhardPoetz)
- enable publishing process (n.n.)
- write the web application (ReinhardPoetz)
  - component that reads from and committs to SVN (n.n.)
  - do the SSL setup of Apache (n.n.)

Roadmap

- I will work on the new Forrest repository and the proposal for the new structure in January. (ReinhardPoetz)
- In February I want to provide a first prototyp for the web application running on brutus.apache.org (ReinhardPoetz)
- In March we should find a home for the web application. AFAIK applications running on brutus.apache.org (the Gump machine) are not allowed to committ into our sVN (ReinhardPoetz)
- Upayavira will invest a large amount of his time into writing and updating docs.

Readers comments

Please use *italics* to add comments above this line, or write more extensive comments and ideas below.

- Tagging documents with simple free-form keywords, as done for links on BD] or pictures on http://www.flickr.com/ is a very powerful yet simple way of providing many useful navigation paths in the documentation. Even if tag-based navigation is not implemented right away, it would be good to add tags (like "sitemap config FileGenerator Generator 2.1.1" for example) when revising existing documents, as opposed to having to revisit all documents later on to tag them. - [BertrandDelacretaz
- There are many hurdles with the publishing process. The main issues are that websites need to be easily recoverable after a potential calamity (hence the current practice of commit generated docs to VCS) and that committers need to be responsible for the various actions (automated is considered bad). Not saying that this is the answer, but see Draft: Proposal for ASF-wide documentation staging and publishing - DC