ExtremeDocumentationOverhaul

Page Summary

- TARGET-AUDIENCE: anybody
- COCOON-RELEASES: 2.1.5
- DOCUMENT-STATUS: draft
- AUTHOR: DavidLeangen
- AUTHOR-CONTACT: dleangen<at>canada.com

What you will get from this page

A proposal for an extreme workover of the current Cocoon website.

Your basic skills

- You know a little bit about Cocoon
- You can read and write English
- You won't throw too many insults our way 😉

Technical prerequisites

- None

Links to other information sources

- Cocoon215TOC

"Project" Members

The contributors to this little "project" are:

- DavidLeangen
- MarkLundquist
- <your name here>

Main Conent

Overview

This page presents a proposal for a complete overhaul of the Cocoon website. Since this approach is very much divergent from the current approach used by the Cocoon project, we expect that it will mostly generate resistence and will not actually materialize on the main Cocoon site.

Nonetheless, this is still a useful excercise to:

- Explain our vision of what the site should be; and
- Act as the basis of a new Cocoon website; or
- Act as the basis of a new documentation project external to Cocoon; or even
- Act as the basis for a commercial project

General Principles

The new website should:

- Make the new content user-centric rather than product-centric
- Have information that is more accessible to new users
- Provide more commerical-friendly content
- Place more emphasis on eduction

To accomplish this, the content should:

- Eliminate the wiki, or have page timeouts
- Be very concise, ruthlessly deleting unnecessary content
- Think in broader terms about who uses the site
- Focus more on what the user wants to accomplish than on what Cocoon does
- <to be continued>

Elimination of the Wiki

We are proposing that the wiki be eliminated, or at least relegated as a type of sandbox.

As a sandbox, each page may exist for a maximum of 3 months before being deleted. The logic behind this is that if the information is useful, then it should be promoted to the main website. Otherwise, the information is not useful and should be removed.

Proof of the need of this is the large number of unused pages on the wiki. Although great effort has been made to clean up the wiki, inevitably, it will again become messy after a few more months of use.

Target Audience

Currently, the site is targeted towards people with programming skills who plan to use Cocoon themselves for development. We believe, however, that this approach is too narrow and does not help to promote Cocoon for mainstream use.

Again assuming that many of our users are commercial, in many cases, if not most, programmers do not make the actual decision as to what product to use. Rather, somebody higher up the ladder determines what product will be used. This person or persons have many factors to consider beyond nice technical design and any final decision is based on perceived risk versus reward, with an emphasis on the word "**perceived**".

Or, a user may be a reporter who has heard about Cocoon and wants to report on it. In many cases, the reporter has little technical knowledge: we need to spoon feed the information. If the reporter cannot understand what we're trying to do, no article will ever get written. However, we want to reach not only developers, but also the decision-makers who's interests are more commercial than technical. We therefore want to make the Cocoon message simple enough to be passed on by the non-technical. As it stands, the information is too technical.

In any case, one fundamental principle holds true: if people cannot relate to the content, if they cannot understand what we're saying, they will simply move on. First impressions are important.

We believe, therefore, that the intended audience must be expanded and the content adapted in consequence to relate better to each of the target user profiles.

Task-based Content

Navigation of the site should be task-based and not content-based. What we mean by this is that content should appear according to what the user is trying to achieve, rather than according to the information we want to present. This means, counter to the fundamental programming axiom of "once and only once", that information may be duplicated. Note, however, that the information source need not be duplicated: only the resulting HTML.

For example, User A is a company decision-maker with moderate technical skills. She has been nagged by some of her developers to look into Cocoon, since they have used it before and believe that it could help their company. Her goal, then, is to make a first evaluation of Cocoon to evaluate the risk of using an open-source project from within the company.

User B is a very experienced developer with fairly advanced skills. User B doesn't want fluff, but wants to get right to the meat. He doesn't like people spoon feeding him information, but would rather see what's inside Cocoon and how it works and make a decision for himself.

User C, however, is fairly new to programming, or at least Java programming. His job is to build a corporate website for a small company of 50 people. He has used PHP, but is frustrated with its limitations. Hearing about what Cocoon has to offer, he wants to give it a try.

Each of these types of users have not only very different goals, but different skill sets. What they want from the site is very different. Using the "once and only once" approach, it is simply not possible (except using some business logic in a dynamic site, which is not what we're proposing here).

<This could be a complete article on its own...>

Proposed Content Structure

This section explains the approach we will take to determine what the structure of the new site should be.

Fundamental Questions

To determine what the structure should be, we first need some way of being able to answer these fundamental questions:

- What type(s) of user(s) will be visiting the website?
- What is the user's goal?
- What, precisely, do we want to communicate?

Use-case Browsing

Most web sites are structured according to what the author's want to say. They look at their product and make a functional decomposition of the site content based on the product's structure.

What we are proposing here is instead to base the structure of the site on the use-case: what does the user want to accomplish?

There are several possible use cases, including:

- Obtaining quick reference information about Cocoon
- Evaluating Cocoon for the first time, which includes
    - Getting a basic understanding of Cocoon
    - Comparing Cocoon with other products
    - Understanding how to "fit" Cocoon in a current environment
- Delving more deeply into a particular aspect of Cocoon

<to be continued>

Discussion Between Mark and David

General

MarkLundquist comments:

(I'm listing this comment first, not because it is the most important – it isn't – but because it's the most general). Just to clarify (and maybe the title of this page doesn't do it justice), the proposal here isn't limited in scope to just the Cocoon documentation *per se*, although it certainly does include that, but really it's about a whole new official Cocoon site, is that correct? See also comment [5] below...

DavidLeangen replies:

Ok, good point! How about we change the title of the page to something like design_of_new_official_website (name intentionally written this was so as not to inadvertently create a new wiki page)?

Indeed, the proposal here goes beyond just the documentation. It is intended to also help with the communications aspect of the project. I feel that from a marketing perspective, much work needs to be done. The long-term success of Cocoon will, IMHO, also depend on its wide-spread adoption in industry, and not just on the core community.

I believe that such an approach is not necessary with, say, Apache Server. Why? Because it is not a product aimed at the "general population", so to speak. A sysadmin will install it once and, well, that's pretty much it. Cocoon is very different. In an enterprise, several people will be involved in working with Cocoon on a daily basis. The way the company evaluates and uses the product is much different. So, just because things worked well at Apache until now with other projects (such as HTTP Server), that does not mean that the same approach will work for Cocoon.

Elimination of Wiki

MarkLundquist comments:

+1 on all the desiderata listed under *"General Principles"*, and also your proposed approach...

...with the notable exception of *Eliminate the wiki*.

The Cocoon project needs a Wiki and probably always will. The Wiki is a good thing! Its use as front-line documentation is arguably bad, but it also arguably represents a current best effort (it's better than no docs at all!). Your proposal w.r.t. the wiki approaches it from the wrong end. We don't have poor real docs because we have docs in the wiki, we have docs in the wiki because we have poor real docs. The idea, IIRC, was to use the wiki as a staging area for new doc material. That seems like a sound approach, so let's just (a) make sure that the new docs do what we want them to do (this relates to all the organizational and content principles you've outlined, and (b) finish the job, i.e. makes sure the good stuff gets promoted from the wiki to the real docs. It's possible that one reason that proto-docs have languished in the wiki is because of dissatisfaction/uncertainty about the lapsed state of the official docs, e.g. "this is good stuff, but where should it go?", or "lets wait until the doc overhaul to merge this in".

Striking elimination of the wiki from your proposal will correct its scope and make it more acceptable.

Once the real docs are up to snuff, the original wiki pages should be replaced with notices that say "this page has been superseded by <this> on the Cocoon site. Please go there for the most current information for Cocoon 2.1". We do that rather than deleting them, so that people who have wiki pages bookmarked have somewhere to go. Insiders doing archaeology on the docs can always look at historical versions of the wiki pages.

DavidLeangen replies:

I generally agree with your comment. I had never thought about in this way before and I appreciate your insight.

How about a slightly different approach, then? First, let's agree on the goals:

- we want to use the wiki as a kind of "incubation" area for new information
- useful information should be incorporated into the main website
- "successful" pages will be replaced by a reference to the new page in the website
- "unsuccessful" wiki pages should be kept for historial purposes
  I agree with all of these goals. (Except I am worried that the reference used for a "successful" page will soon be outdated--and therefore quite useless.)
  I further propose:
- the wiki should *not* be used as a main source of information
- the wiki should only be used as a source of information for those really pushing the boundaries of what Cocoon can do and is intended for
- the wiki should not be "endorsed" as a source of information

- information on the wiki should not be supported (if it is, then it belongs on the official site!)
- we need a mechanism to "outdate" old wiki pages
  So, with these two sets of goals in mind, one approach we could use is, for each new release of Cocoon, *all* wiki pages get archived. Above in the proposal, I propose deleting a page after 6 months, but this conflicts with the goals we just set here. By archiving all files after a new release, we would ensure that:
- pages are no longer out-of-date (authors or interested persons will move a valid page back to the current wiki area if the information is still valid)
- the wiki remains clean and useable for its intended purpose
- the wiki does not become the main source for consulting Cocoon info
- we keep information for historical purposes
  I think that we (and many others) agree that, while the wiki is useful, it has become a problem. I believe that we need to take a radical approach to solving the problem. Perhaps eliminating it all together goes too far, I agree. However, I think that we should nonetheless include *some* type of solution.

User Profiles

MarkLundquist comments:

It seems to me that your example document users A, B & C are probably in fact exactly the most common profiles, and we should consider them to be the actual target profiles for this effort.

Deprecation of Documents

MarkLundquist comments:

W.r.t. the "ruthless deletion" (+1!)

I think this should start with everything that doesn't represent current best practices.

Whatever is deprecated or clearly on the path for deprecation should be annexed off so that it is clear that it no longer represents current thinking. The last time Cocoon docs got the level of effort they deserve was for "old Cocoon", which means that Actions, XSP & Logicsheets, and to a degree the idea of custom sitemap components, all still figure prominently. It may be that one reason that stuff hasn't been touched is the feeling that "new Cocoon" is in certain respects not yet "ready for prime time", or at least that its documentation isn't. But that is a bad think. The best that can happen with such an approach is that users eventually figure it out anyway, but only after wasting a bunch of time trying to learn obsolete concepts. It's time to be honest in the docs. Yank everything that's only relevant to "old Cocoon", and if there's nothing yet to replace it, we should just be honest and say something along the lines of

- Cocoon has a really cool system for this, which we call 'X'. Unfortunately we haven't managed to get some documentation written for it yet, so in the meantime you can check out these resources:
  - [whatever samples]
  - [wiki articles]
  - [list archive references]
    Or,
- There isn't a great way to do X right now that doesn't require writing some Java code. We're trying to figure out a better way; in the meantime, if you have some Java skills, here's how you do it: <link to wiki page or whatever>.
  The current docs give the impression of Cocoon being more complete and better documented that it really is, and maybe we don't want to lose that "impression"... but we do users a disservice by hanging onto the obsolete stuff. We have a "double-hump" technology situation here and we're still a bit in the "gap" betwen old Cocoon and new Cocoon, but I think we should still be as accurate and honest as possible.

DavidLeangen replies:

That makes sense. The product is always in a state of evolution.

I think that we need to separate what is "cutting edge" from what is "standard cocoon". If we consider a classic bell curve for the evolution of a typical product, the vast majority of our potential users do not want to live on the cutting edge. They want something that is proven, stable, and reliable.

Another point that bothers me is that there are so many ways to do one thing. I have brought up this issue before and quickly learned that several people believe this to actually be an advantage of Cocoon. It is not my place to say what is best, but I do strongly believe that for the user profiles we are targeting, by far the best approach is to have a "standard" set of best practices.

Ok, the above topics diverge somewhat from your comment, so to reply directly: if the feature exists, then it needs to be documented on the webpage. Period. Now, the problem is: who will do this? Nobody wants to do documentation... so how do we pull this off?

Look and Feel

MarkLundquist comments:

This comment is probably most relevant to your "User A" profile, the corporate manager.

Visual design communicates a message. For the current Cocoon site, that message is, "Made for geeks, by geeks". It has that tab-y, Forrest look. The Apache branding is very strong. What's wrong with being strongly identified w/ Apache? Absolutely nothing. So what's wrong w/ the Apache branding? Just that it *happens* to be kind of sucky-looking, that's all!

Compare cocoon.apache.org w/:

- orixo.com
- wyona.com
- orbeon.com
  So assuming we had a better design in mind, what are the tactical details for implementing it?

DavidLeangen replies:

It's true the the Apache brand is something worth building on. For the rest see my comment above concerning the nature of Cocoon vs. other successful Apache projects.

JonathanMerriman replies:

Major design issues with the site:

- The heading and navigation take up too much real-estate.
- Page body is too wide.
- Page layout seems unbalanced, fragmented.

Quick fixes:

- Reduce padding of the page body.
- Use a narrow, serif font such as Times for the page body.
- Redo the masthead. No use saying "Cocoon" twice.
  I am actually quite fond of the Struts site (and the main ASF site, for that matter). Another example of a good site layout is "The Mono Project". It has a clean layout that nicely separates tasks (general info, developing, and contributing, with quick links to downloads and documentation).

Random Thoughts

MarkLundquist comments:

[hmmm... I have some thoughts but I have to close this post out for now. To Be Continued...!]

DavidLeangen replies:

Ok, I look forward to hearing about them! 🙂

Thanks for taking the time to comment on all this stuff.