Status/RoadMap

- Where are we?
- Where do we want to be?
- Who is working on what? (Priorities/goals/majors issues/roles)

TikiTeam

Who is working here generally? Link UserPage.

Trackers

- Bugs
- RFEs

Knowledge Base Variant of CMS

- tech support
- patches

Competition and standards

List of other products with similar/interesting/related features.

Document Management

Knowledge Management

- ITMS: Topic maps in content management — Discusses CMS systems that go beyond tree based content organization to knowledge organization. Think of content metadata being stored in the topic map rather than disconnected across multiple documents; and think of the superior "findability" of content with a topic map.

Here I would like to see some "editorial" content. How do our features compare to others?

CVS Doc section

This is where new features being developed and only in CVS are documented. When the CVS becomes RC/official release, the info in the CVS docs is transferred to update the official docs (FeatureXDoc).

Discussion/participation

Where ideas can be exchanged, debated, etc. Interested people can subscribe to the wiki page and/or to these forums as they would a mailing list.

It is noteworthy to distinguish between *document management* and *knowledge management*. Document management usually provides features such as hierarchical categorization, access control, check-in check-out and version control, and some metadata (author, last modified, organization, etc.). Knowledge management (knowledge store, knowledge base) may provide the above features, but is principally focused on creating intelligent associations between documents (content objects) so as to increase the *findability* of content, to facilitate *information discovery*.

For example, a support website can store a bunch of documents, tech notes, comments, emails, announcements, etc., which a user can search via keywords. The user's potential for success will depend on his judicious choice of search terms and his existing level of knowledge of the subject--a level that by definition will most often be low. Such a user will fail to find many relevant resources.

A navigable topic directory is a first step to help you discover what you don't know. But taxonomies (category structure) and ontologies (category naming) are burdensome to develop and don't capture perhaps the majority of the knowledge that is available about subjects. Here is where the *associations* of *topic maps* come in. *Associations* allow you to define relations between content objects. They provide the "See also" function of back-of-the-book indexes. An Integrated Topic Map Management System (ITMS) can generate web pages with rich context-sensitive navigation links to related information for each content object and each category and category level. Such context-sensitive navigation functions like a directed graph, leading the user from one topic to a related one, be it going hierarchically deeper (more specific), broader (less specific), or toward other relations (i.e., see also).

With this approach, you discover things just like you discover things when you look up something in the encyclopedia index. But, with a topic map generated web page, the index (a contextual subset of it) becomes the trimmings (navigation) of the page--it's always with you. Through a combination of keyword search (for initial narrowing of a search), hierarchies, and associations, a powerful "information discovery" system can be created.

Notice that this approach also removes the "walled-in" nature of an application. If the all of Tiki is built with topic map navigation, the user might smoothly flow-in to the knowledge base and smoothly flow-out without ever really feeling like he went in or out of anything. Since knowledge is essentially throughout a website, I suggest thinking of the whole website (and any website of any type) as a *knowledge base*. *--pacoit*

Coofer Cat:
I've got the seed of an idea of how to actually implement this sort of thing. My original prototype is not necessarily incompatible with it, but is severely limited in comparison. I'm semi-on the case with some development of this, because I really want it (!). I've got a bit of SEFURLs work to get out of the way first, but once that's done I'll get going.

I'm thinking that I'd create a series of documents in the database. Each one is numbered, because I want to be able to reference them by ID rather than anything more complex. Each document has a connection to it's own linked list of topics. These are the topic map for the document. The linking of topics by a document means that topics become related.

From the finding a document view point, a user can start with any topic. That topic will then have links to numerous other topics, which are linked further, and so on. Each topic also has documents associated with it. Thus, a document appears in multiple places in the topic map.

So, if you have one document that has topics "Tiki", "PHP", "Mysql", and another that has topics "Mysql" and "SQL scripting", you would have a topic map with four possible "start points". If you started with PHP, you'd have options to look at the first document, or proceed to topic Tiki, or Topic Mysql. Proceding to Mysql would have a link to document 2, and a link to SQL scripting. SQL scripting only has a link to document 2, and presumably a link back to Mysql.

I really like the sound of this sort of thing. It has the danger of ending up with a horrible spaghetti of topics that have very few documents in each topic. As a result, it seems a limit on the number of topics for each document might be a good thing (configurable, of course).

I think also that documents should have some sort of "type" attribute. That is, an attribute that says "wiki page", "cms article" etc. That way, one could navigate all wiki pages, viewing a subset of the topic map.

That's my two pence worth. It's the start of a real implementation design, which sort of implies database structure. Its already well ahead of anything I ever did with my prototype (very small amounts of data design went into that!).

...Coofer Cat