Architecture Suggestions From Mozilla

Below is a list of suggestions from our friends at Mozilla. Support.mozilla.com has huge load (Over 10 000 000 page views per week) and have concerns about performance, security, etc.

They are using a snapshot (modified with various enhancements and fixes) of BRANCH-1.10, approx revision number 10409 (Nov 22 2007), while current development version is at 21000.

SUMO code base:
http://viewvc.svn.mozilla.org/vc/projects/sumo/

The 1.10 code used by Mozilla was not a release version but a development branch. That branch was later renamed and released as 2.0 (2008-08-03).

Tiki 3.x was released on 2009-05-19 and 4.x is coming up shortly.

This page was originally at: https://wiki.mozilla.org/Support/TikiChanges
Related:

- SUMO Upstream Process
- https://wiki.mozilla.org/Support/SUMO-Tiki-Collaboration

This is a list of what we see as the things in Tiki that need changing/fixing - the focus is on things about Tiki that make it harder for us to extend it. We'd like to know which of these things have been improved in the current version and which ones would still be bothersome.

The items most critical for us are marked in bold.

Missing critical features

"multiple databases at once" as in replication, sharding, etc We have built workarounds for these issues.	please add link to patch/code Perhaps providing VM images with configured servers would help test.
"multiple web servers" as in having lots of machines serving tiki - the specific issues there are around e.g. file uploads, sessions, etc. We have built workarounds for these issues.	please add link to patch/code
No built in memcache support	Let's add to Tiki 4. patch/code or perhaps look at Zend_Cache This has to remain optional, but replacing the implementation in cachelib with ZF could help the abstraction
How are sessions implemented in Tiki 4? We have put them in memcache since they were in DB before, also didn't like the default for auto session start (too expensive)	please add link to patch/code Once again, must remain optional. Many features in tiki depend on sessions for better or worse.

No unit tests	Some unit tests Long term, but nothings stops from writing tests. Code has been organized to contain PHPUnit tests from 3.0
No well defined "core" set of functions or library. Frequently used functions sometimes lie in different files.	To be discussed
Missing essential functions. g.g. for base url - I wouldn't trust using $base_url (global variable may be changed), for notifications (see also Other), sanitization (e.g. make_plain or something).	Feel free to add them as needed.
Overuse of global variables - makes it hard to trace what comes from where. I think it's critical, we should eradicate use of globals.	Impractical on the short term.
No obvious hook system or similar makes it hard to add functionality cleanly	Significant work required to preserve all current functionalities and to refactor them into a hook system. This would be desired, but it's a large task.

Scalability issues

Includes of thousands of lines of unnecessary code on every request, code should be included on a more as needs basis or autoloaded	Most of it is loaded when required. tiki-setup.php is still large, but it got organized since 1.10. Problems exist with monolithic libraries like tikilib that contain too many functionalities. Significant work is required to move code where it really belongs.
No apparent benchmarking (see also security)	Done occasionally, but not systematic. Lots of improvements were made in 4.0.
Too many layers of includes. E.g. tiki-index.php includes tiki-setup.php includes tiki-setup_base.php includes tikilib.php includes tra.php... etc. These make it very hard to track where stuff is coming from.	grep...
Template system is really slow — Smarty is slow as it comes, but with Tiki filters added on top it you can really feel the slowness	We would like to try Smarty 3 Removing smarty is unlikely. Filters would have to be verified, but otherwise, smarty is compiled into straight PHP. Most of the templates are very complex and generate bad HTML. This can be improved. Producing less output will speed things up. Some improvements were made on this front, but this will take a long time at the current rate. report of better performance in Smarty3

Security

Top level security code really slow (version from Tiki 3) and previous versions have not worked correctly (e.g. ur&lt;x&gt;l)	filters have been deployed to many often used plugins but more deployment is needed
Escaping in templates not enforced, scattershot	
File upload code has caused problems in the past (fixed by us, did the patch make it into Tiki?)	Please add link to fix/patch

Code organization

Inconsistency. This is everywhere. Naming conventions, mixed approaches, etc, etc.	Not going to change any time soon. Sorry.
Modules (code stored in db) both a security issue and a version control issue	We don't understand, can you elaborate?
File naming and organization could do with work **hundreds of files in the root dir, mostly starting with tiki-. **libs not consistently located (code layout issue) **some libs lack cohesion	
Really long functions	
Uncommented/unphpdoced/unusefully commented functions	
Poorly indented code	Improving. Re-indenting code is now allowed by project guidelines as long as it's made as an isolated commit.
Messages mixed between php and template files, e.g. error messages assigned using tra(). This should never be used since it creates issues with portability and consistency (see https://bugzilla.mozilla.org/show_bug.cgi?id=510282). Also makes it hard to track where messages are coming from.	Errors are pretty much the only case of this. Messages are translated where they are written so the scripts can pick them up.
Mixed implementation in include files: classes/objects versus stand-alone functions. This means that when you're including a library file you have to remember which of the two it is. Also, some of the choices to use one vs the other seem arbitrary to me, and in some instances use of globals is abused versus in others using objects. (if need be I can find concrete examples)	
Lack of coding standards. A really annoying example: I can remember a situation when a PHP file had spaces after the closing tag and it took me a long time to figure out that's what caused a page not to load. Coding standards should include no closing tag.	DevTips All closing tags have been removed.

Database

Lack of critical indexes	Indexes are added as needed. Please report indexes that
Prevalence of slow queries	These issues only come up in large installations with high volumes. Improvements are always welcome
Confusingly named tables (e.g users_users)	
Use of MyISAM when InnoDB should be used	Should be option Only thing holding this back has been the search. Hopefully this one will be resolved as part of 5.0. InnoDB should really be used, but questions remain with the impact on existing installations.

Localization

No support for placeholders makes it almost impossible to translate strings with variables	Placeholders have been added in 3.0.
No support for markup (just plain text) — this is connected to previous	HTML is used occasionally, although not encouraged.
No caching, at least none that I'm aware of. This + smarty + loops with localized strings almost certainly leads to ridiculous page load times.	Translations are cached for the templates. The template is compiled on a per-language basis in templates_c. It remains an issue with the strings in the PHP files and the tra() function is rather slow with the volume of strings in tiki.
Why doesn't Tiki use po files?	PO Convertor for TikiWiki The exact answer to this is that no one did it. Discussion: Should we change from language.php to native .po files?

Other

Out of date database libs (no support for stored procs etc) (note I think this is solved by the change to PDO)	Stored procs are not used on purpose because they prevent database abstraction. At least, this was the case before. Imposing a limitation to MySQL 5 is not much of a problem anymore. ~~However, with the come-back of PgSQL, writing procs twice is impractical.~~
No centralized notification system. Dumping errors to error.tpl does not allow notifications on page. No types of notifications: warning vs error vs inform user messages	This could be implemented. It just requires additional work.
No built in pagination support (which other CMSes have)	Added.

No centralized form handling. It feels very unsafe to build forms in Tiki.	This is an issue. Filtering now allows to specify filters at a single location, but it's still not a complete solution. Feel free to propose/implement something, but I have generally been unsatisfied with form libraries.
Abuse of $_REQUEST variable	
Some features not well documented	ML: Please let me know when/where you need help?
A way to provide redirects would be useful. E.g. Drupal has ?destination parameter that redirects to a different location after form submission. Lots of places in the code have the header('Location: ') tag and this + lack of base_url function leads to nightmares when wanting to change redirects or figuring out why this page dies or goes somewhere, etc.	Please go right ahead and add!
Use of 'y' instead of booleans.	Permissions are done partially. Need more people that have time to do the rest.
No real "modular" system. Files from different modules are mixed together, and files from the same module must be kept in different places. Also applies to themes. To add an on/off switch for a module, must edit multiple files (and first find out which ones need editing).	
Use of Smarty	Overall, we are quite happy with Smarty. With time, we are using Smarty more & more and are creating plugins to make it easier to harmonize user interfaces. Changing would be an unthinkable quantity of work (it's been the template engine since the beginning of the project). Smarty 3 is almost in Beta, how about we get in touch with Smarty developers and collaborate to address concerns?
Lack of menu system. Adding items to menus requires editing templates.	menu
Disorganized/unclear admin system. Under "General" is a (currently unused) option called Browser Title, which is akin to "Site Name," while under Look &amp; Feel are options to change things Site Logo Title and Site Logo Alt description, which could be combined into more logical or familiar controls like "Site Name" and "Site Tagline." (It also says you should go to Site Identity, which is blank.) It seems like the poor organization of the admin system, particularly relating to themes, lead to it being ignored.	Site identity was a newish at that time, and things were cleaned up for 2.0, and improved in 3.0 and 4.0

Links used as headers in admin section. For example, in Look &amp; Feel, instead of &lt;h#&gt; elements, it uses &lt;a href="#"&gt; to separate sections. At best, the links are links to themselves, at worst, to the top of the page.	4.0 will bring major changes to the admin section if deployment gets completed.

Examples

Refer to http://pastebin.mozilla.org/667336

Mixed sources of information. There are 7 lines of the form if ($prefs['something'] == 'y') followed by an if (isset($_REQUEST['somethingelse'])) then another reference to the global $prefs then one to if (!empty($_SESSION['something'])).

Mixed responses to information. Most lines in the snippet include a file. Not all of them do. Some take additional actions.

Summary

- **Performance issues** These can generally be solved on a case by case basis. A few hours, profiling can be made and the biggest problems resolved. There are a lot of low hanging fruits and everyone would benefit from it. It's just a matter of testing with realistic data. Spending half a day or a day per week to improve performance from trunk rather than the 1.10 fork would ensure long term improvements.
- **Code organization** This is not going to change soon. It's years of work to clean up and very few are remotely interested in doing it. There are loose conventions in the code and you get used to it. Clean-up will be made if only an attempt to improve/refactor the code is made before hacking in new features. Following the commits and providing guidance to less experience developers can also help.
- **Security** Either audiors gave up or it improved. Just like performance, this needs attention. There is no silver bullet.
- **Scalability** Mozilla is the first installation to ever attempt to scale beyond a server. Contribution and experience is welcome.

More

From James