FLOSS Web Application with the most contributors with direct commit access to the whole code base

Note: this one, I am not sure. Let's do some research to see if this true. (and if not, let's learn from that other project
wikiquote.org wrote:

> *The problem with Wikipedia is that it only works in practice. In theory, it's a total disaster.*"

tiki.org wrote:

> *Software made the wiki way*

Tiki is the Free/Libre/Open Source Software Web Application with the most contributors with direct commit access to the whole code base.

- FLOSS -> http://en.wikipedia.org/wiki/Free_and_open_source_software
- Web Application -> http://en.wikipedia.org/wiki/Web_application (This includes Wiki, CMS, Groupwares or any application comparable to Tiki)
- contributors with direct commit access to the whole code base: The Tiki model is to encourage everyone to commit to trunk directly, wiki way and improve what they need improved, without gatekeepers. We trust that people will ask for help when they feel they need it. Everybody can watch and revert every single commit. In practice, if someone has a concern about a commit (introduces a bug, duplicates functionality, bad design, etc.), it leads to a discussion and a compromise is found, and likely an additional commit to address the issue, or a revert.

Tiki has over 500: http://info.tiki.org/article188-Tiki-reaches-500-contributors-with-commit-access

The second place seems to be Plone, with 450 core contributors (but do they all have commit access to the core and whole code base?)

Some other projects have overall more committers if you count extensions, but they are not direct committers to the core. For example, in Drupal, there are core maintainers, who approve the commits.

Why is direct commit access important?
Direct access to the whole code base is important because it makes it easier to refactor the code and not be limited to what the core provides (say you are developing an extension and can't modify the core code)

While the idea of each commit (code contribution) being reviewed is appealing, you need a lot of active reviewers and it adds complexity and delays. In large projects, you end up with hundreds or thousands of yet-to-be-accepted commits. And because these unaccepted commits can be in a queue for several weeks or months, they can, at one point become invalid because another commit breaks them.

But no gatekeeper doesn't mean there is no review. Since

- We dogfood Tiki for *.tiki.org sites
- We have Pre-Dogfood Servers that are upgraded 4 times per day, and can show us what our *.tiki.org would be like if we upgraded today.
- The commit mailing list has over 80 subscribers, many of which check every single commit and can react if something looks risky. The whole community is potentially affected by each commit and has a direct incentive to check stuff.

We want the "The Simplest Thing that Could Possibly Work" and avoid this dependency-hell between unapproved commits. Besides, since the vast majority of commits are just fine, reviewing each commit is taking resources away from regular development.

See also

- FLOSS Web Application with the most built-in features
- FLOSS Web Applications with the fastest release cycles
- FLOSS Web Application with the longest predictable security support period

alias

- Free and Open Source Software Web Application with the most contributors with direct commit access to the whole code base
- FOSS Web Application with the most contributors with direct commit access to the whole code base