ChiliProject - Task # 30: Code review of latest Redmine commits (November and December)

Status:	Closed	**Priority:**	Normal
Author:	Muntek Singh	**Category:**	
Created:	2010-12-29	**Assignee:**	Eric Davis
Updated:	2011-02-10	**Due date:**	
Remote issue URL:			
Description:	Code review of latest Redmine commits (November and December)		

History

2010-12-29 04:51 pm - Eric Davis

I've been reading a few of the commit and there will be a few that need to be reverted or reworked. Commit 73167fb4f26b34c1da9c9bd9c12d175442a8e56d is the first questionable one that needs to be reviewed (2010-11-06 03:57).

2010-12-29 04:55 pm - Eric Davis

- Target version deleted ()

2011-01-02 05:49 pm - Eric Davis

I'm thinking we should create an issue for each commit to do a "code review". This might be useful later on too, that way there can be discussions around each commit and the reviewer can be recognized ("Closed/Reviewed by Eric").

2011-01-12 02:29 pm - Eric Davis

- Tracker changed from Bug to Task

2011-01-13 03:37 pm - Eric Davis

- Subproject of deleted (#28)

2011-02-01 08:42 pm - Holger Just

- Target version set to 1.1.0 â€" Bell

2011-02-01 10:49 pm - Felix Schäfer

- Status set to Open

2011-02-06 01:58 am - Eric Davis

- Assignee set to Eric Davis

Starting on this

2011-02-06 05:12 am - Eric Davis

Did some fast code reviews of everything up to 1.0.5 on [[edavis10|my page]]. I've accepted everything so far but I found quite a few things we need to make a decision on (e.g. keep, revert, modify). I suspect the next set will have more things.

I'm still not sure how far we should go for our 1.1.0 release. Should we be mostly-compatible with Redmine 1.1.0?

2011-02-06 05:14 am - Eric Davis

P.S. Using this branch https://github.com/edavis10/chiliproject/tree/ticket/master/30-upstream-code-review

2011-02-06 12:26 pm - Felix Schäfer

Eric Davis wrote:

> I'm still not sure how far we should go for our 1.1.0 release. Should we be mostly-compatible with Redmine 1.1.0?

I think that would be good at least for the DB to ease upgrading/changing back-and-fro, I don't have that much reserve regarding the external APIs though.

2011-02-09 12:17 am - Eric Davis

Okay, I'll keep going up to the Redmine 1.1.0 release and see what else I can pull out of 1.1.1.

For some of the larger changes, we might have to make some branches and cherry-pick. There are a lot of SCM changes that we might want.

2011-02-09 01:07 am - Eric Davis

- Status changed from Open to In Progress

I've finished the code review and it looks like most of the commits will be fine to include in ChiliProject.

* There are quite a few we need to discuss
* There are some that need bugfixes for
* There are two commits I want to completely reject (User deletion) (Search on [[Edavis10]] for "rejected commit" to find them)

I propose we:

# take all of the commits up to Redmine r4725 (ced782ecb21c62)
# skip over the to commits to reject
# cherry-pick the rest of the commits
# open issues here to discuss and/or fix the commits I've flagged
# after a discussion we can come back and revert or just change any code needed

If we do it this way, I can do some merge magic and make our repository start right at r4725. This should make upgrades from Redmine 1.1.1 painless and also get some of the great stuff from Toshi MARUYAMA.

2011-02-09 01:14 am - Robert Chady

This seems like a valid approach. It would be nice to get user deletion working, but not the way it is in Redmine.

I am also glad to hear there will be what sounds like a formal code review process in place for CP.

2011-02-09 04:33 am - Muntek Singh

> I propose we:
>
> # take all of the commits up to Redmine r4725 (ced782ecb21c62)
> # skip over the to commits to reject
> # cherry-pick the rest of the commits
> # open issues here to discuss and/or fix the commits I've flagged
> # after a discussion we can come back and revert or just change any code needed
>
> If we do it this way, I can do some merge magic and make our repository start right at r4725. This should make upgrades from Redmine 1.1.1 painless and also get some of the great stuff from Toshi MARUYAMA.

Looks good, I glanced through the review as they came and didn't see any I didn't agree with you, nor did my programmers find any. This proposal is sound, and I look forward to seeing it.

2011-02-09 08:36 am - Felix Schäfer

Eric Davis wrote:

> I propose we:

>

> # take all of the commits up to Redmine r4725 (ced782ecb21c62)

> # skip over the to commits to reject

> # cherry-pick the rest of the commits

> # open issues here to discuss and/or fix the commits I've flagged

> # after a discussion we can come back and revert or just change any code needed

>

> If we do it this way, I can do some merge magic and make our repository start right at r4725. This should make upgrades from Redmine 1.1.1 painless and also get some of the great stuff from Toshi MARUYAMA.

Works for me.

2011-02-09 08:14 pm - Eric Davis

Thanks everyone, I'll redo my branch and get it ready to send to the main repo.

2011-02-10 01:22 am - Eric Davis

- Status changed from In Progress to Closed

Alright, the main repo how now been updated and we are ready to get started on actual development for 1.1.0 now.

h2. Gory details for git folks

# First thing I did was to remove my feature branch and re-branch from the last master version (commit:a52417eca9311aabaeaab847873aea52cb78ce52)
# Then I found the last liner commit we wanted to use, which was commit:ced782ecb21c62
# I make a local branch off of that commit (@git checkout -b redmine-r4725 ced782ecb21c62@)
# Then I did a *fast forward merge* (git's default) to make my feature branch look like it started from there. @git checkout ticket/master/30-upstream-code-review && git merge redmine-r4725@
# Then I deleted the @redmine-r4725@ branch
# Then I created a new branch so I could pull out the commits we want to keep but skip over the ones to reject @git checkout -b ticket/master/30-upstream-code-review-cherry-pick@
# Using git format-patch I created a patch for the single commit we wanted to keep that was after the rejected one (@git format-patch e17fadd07ac34f96a10a8b1e77b9ee06bc4149b2^..109fd2cdfc880a1b91bf6e658e4bc35c6bdcf85f@)
# I applied this patch (remember I'm on the -cherry-pick branch now) @git am < 0001-Do-not-show-for-only-project-I-select-notification-o.patch@
# Using the same process, I created a single patch that was a combination of all of the remaining commits we wanted to include @git format-patch e17fadd07ac34f96a10a8b1e77b9ee06bc4149b2..109fd2cdfc880a1b91bf6e658e4bc35c6bdcf85f --stdout > full.patch@
# Then using git am I applied all of the commits in a single batch run (@git am full.patch@)
# During this process several patches failed and I had to merge them by hand.
# Once this was done I then merged the -cherry-pick branch back into the main feature branch, this time making sure the merge *wasn't* fast forwarded so we can track it later on (@git checkout ticket/master/30-upstream-code-review@ @git merge --no-ff ticket/master/30-upstream-code-review-cherry-pick@)
# Using git diff I was then able to compare the final results against Redmine's code. Our branch was identical except for the rejected commits and some whitespace changes in coderay (probably from svn/git). @git diff redmine-upstream/master@
# At this point I checked out ChiliProject's master branch (named @upstream@) and merged in my branch. @git checkout -b upstream-master upstream/master@ and @git merge --no-ff ticket/master/30-upstream-code-review@
# Finally after a quick sanity check, I pushed this up to github into the master branch @git push upstream upstream-master:master@ (I also configured this mapping to be automatic in my local config)

I know this process looks confusing but we will probably never have to do it again. Future pulls from Redmine will be less work and we can cherry-pick as we go too.

2011-02-10 05:19 am - Toshi MARUYAMA

Mercurial Transplant extension is very smart.
http://mercurial.selenic.com/wiki/TransplantExtension

2011-02-10 12:01 pm - Simon HÃ¼rlimann

Mmh, why all the "magic"? I tried to reproduce by simply using the latest redmine version and just revert those commits you don't like. git diff to master seems to show that this resulted in same tree.

Beside being simpler, the reason for this is that git cherry doesn't return an ever growing list of commits. It also allows to do more simpler merges in the future...

just my 2c

Good work anyway!

2011-02-10 12:13 pm - Simon HÃ¼rlimann

Another point: git revert XXX does clearly show the intent of reverting this patch. And the commit don't all get edavis10 as commiter;-)

Just to make sure I correctly understood what you did: Am I correct that what you did is more or less a 'git rebase -i XXX' where XXX is the redmine/master with dropping those 3 commits?

2011-02-10 11:31 pm - Eric Davis

Simon HÃ¼rlimann wrote:
> Mmh, why all the "magic"? I tried to reproduce by simply using the latest redmine version and just revert those commits you don't like. git diff to master seems to show that this resulted in same tree.

To have a clear history of where we branched at. Look around Feb 3rd on the "network graph":https://github.com/chiliproject/chiliproject/network

> Beside being simpler, the reason for this is that git cherry doesn't return an ever growing list of commits. It also allows to do more simpler merges in the future...

I'm not sure I understand, could we talk about this in the forums? I'm interested to see if there is a better way in the future.

> Another point: git revert XXX does clearly show the intent of reverting this patch. And the commit don't all get edavis10 as commiter;-)

I didn't want to put one of our revert commits in the middle of the history and I didn't want to revert them at the end.

> Just to make sure I correctly understood what you did: Am I correct that what you did is more or less a 'git rebase -i XXX' where XXX is the redmine/master with dropping those 3 commits?

Yes, it is almost like that. If I recall @git rebase -i@ would have to rewrite the commits when changing the history and I think it would make me the author and committer. I also think it would reset the commit/author dates too.