Support Platforms - Feature #694

Write instructions/policy for issue handling

02/01/2011 08:58 PM - Teemu Murtola

Status:	Feedback wanted
Priority:	Normal
Assignee:	Rossen Apostolov
Category:	Redmine
Target version:	

Description

I think it would be useful to have written instructions for things like

- What's the purpose of the different issue states? E.g., what's the difference between Resolved and Closed, and when should one use one or the other?
- How should the "Assigned to" field be used? When should it be cleared, when should new issues be directly assigned to someone?
- When should the issue status be changed? One approach would be that whenever the responsibility of the person in the "Assigned to" field changes, the state should also change to reflect the responsibility of the assignee.
- What transitions are allowed between the states and who is allowed to do what?
- How to report version information for bugs?

There are probably others as well. Of course, in addition to writing the instructions, the policy should first be decided on.

Related issues:

Related to Support Platforms - Feature #687: Add content on the front page	**New**	
Related to Support Platforms - Bug #836: Linking to Redmine issues from commi...	**Closed**	**11/12/2011**

History

#1 - 02/02/2011 12:39 AM - Mark Abraham

I think "resolved" should mean that somebody knowledgeable thinks they have fixed the issue, somehow. "Closed" should happen when there's been a pause for further feedback. "Feedback" should indicate that input from people is required (e.g. more data from a bug poster, or a community poll).

Bugzilla-style "works for me" and maybe "feature, not bug" might be reasonable to add.

#2 - 02/04/2011 06:42 PM - Teemu Murtola

Mark Abraham wrote:

> I think "resolved" should mean that somebody knowledgeable thinks they have fixed the issue, somehow. "Closed" should happen when there's been a pause for further feedback. "Feedback" should indicate that input from people is required (e.g. more data from a bug poster, or a community poll).

That's also my idea of the difference between "Resolved" and "Closed". It should also be noted that "Resolved" is still considered as an open issue by Redmine (by default), so it should require some further action. For the "Feedback" state, I think that the default idea is that it comes after "Resolved" if there is some additional information to consider. This could of course be changed, or different states added for different feedback states. But on the other hand, the number of states should probably be as low as possible to keep things simple.

> Bugzilla-style "works for me" and maybe "feature, not bug" might be reasonable to add.

For these purposes, there's a "Rejected" status in the default configuration of Redmine, but by default, only a manager is allowed to put issues into this state, so probably others will never even see it.

Added a few more points to the description, in particular concerning the privileges of different users.

#3 - 04/12/2012 08:58 PM - Szilárd Páll

Bump!

This is quite important so we might want to try discussing it on the mailing list as these issues seem to be ignored by the wider developer-base.

#4 - 04/13/2012 02:57 PM - Szilárd Páll

I'd suggest adding the "confirmed" status for bugs which would differentiate between new/unconfirmed and confirmed bugs.

#5 - 06/17/2012 12:55 PM - Teemu Murtola

- Category set to Redmine

Trying to list some possible situations that may merit their own issue state. The list is long on purpose; we may not want all of these, but I think it's better to think the alternatives through. And even if we add several new states, we probably want to allow for simplified workflows for simple cases where communication of the current status is not important.

- Newly submitted issues naturally get a *New* state. No reason to change that.
- As Szilard suggests, *Confirmed* could be useful for bugs. A more general name could be *Accepted*: for bugs, this would mean that the bug has been confirmed and is deemed worth fixing, for other issues that it is seen as a useful thing to do at some point.
- If the reported bug is not a bug, or a proposed feature is not suitable, or for other similar situations, a *Rejected* state is useful. There is already one.
- When work is being done on an issue, *In progress* is suitable. No reason to change that.
- If the issue needs some feedback/discussion to proceed, a separate state would be useful. *Feedback Needed* is one possibility.
- Possible alternative for the above is a more general *Blocked* for any issue that is waiting for some other action (e.g., a fix for another issue).
- For the two above states, when the discussion is complete or the blocker has been resolved, the issue should move to some other state.
- When the implementation for the fix/feature/task is done and submitted to Gerrit, it could be useful to have a separate state. *Fix Submitted* is one possibility, but this may be a bit confusing as the "submit" button in Gerrit moves the issue out of this state.
- When the implementation is approved in Gerrit and merged to the main branch. A possible name could be *Fix Merged*. This could be useful, e.g., if additional testing is needed, for example by the submitter.
- When there is nothing left to do for the issue, it can be *Closed*.

In particular if we want to use Redmine to track all development and not just bugs (which I think would make the project status much more transparent for people outside the core team, and even within), these things should be considered to get the most out of Redmine.

#6 - 06/28/2012 06:46 PM - Szilárd Páll

Just a brief note on the automated issue status changes. Rossen changed the settings recently and made Redmine automatically change the status to closed when a "fixes/closes" note lands in git. Quite obviously, this has not been tested and implemented properly so I'll disable it asap because IMO it's annoying that

- the issue gets closed instead of set to a reserved state (btw, should this be Resolved/Fix merged/something else?);
- when Redmine does this it shows up as "Anonymous" closing the bug which is just fishy and confusing.

Btw anynonymous is Redmine auto-closing the issue ().

For details and further discussion go to #694.

#7 - 06/29/2012 04:00 AM - Mark Abraham

Teemu Murtola wrote:

> Trying to list some possible situations that may merit their own issue state. The list is long on purpose; we may not want all of these, but I think it's better to think the alternatives through. And even if we add several new states, we probably want to allow for simplified workflows for simple cases where communication of the current status is not important.
>
> - Newly submitted issues naturally get a *New* state. No reason to change that.
> - As Szilard suggests, *Confirmed* could be useful for bugs. A more general name could be *Accepted*: for bugs, this would mean that the bug has been confirmed and is deemed worth fixing, for other issues that it is seen as a useful thing to do at some point.
> - If the reported bug is not a bug, or a proposed feature is not suitable, or for other similar situations, a *Rejected* state is useful. There is already one.
> - When work is being done on an issue, *In progress* is suitable. No reason to change that.
> - If the issue needs some feedback/discussion to proceed, a separate state would be useful. *Feedback Needed* is one possibility.
> - Possible alternative for the above is a more general *Blocked* for any issue that is waiting for some other action (e.g., a fix for another issue).
> - For the two above states, when the discussion is complete or the blocker has been resolved, the issue should move to some other state.
> - When the implementation for the fix/feature/task is done and submitted to Gerrit, it could be useful to have a separate state. *Fix Submitted* is one possibility, but this may be a bit confusing as the "submit" button in Gerrit moves the issue out of this state.
> - When the implementation is approved in Gerrit and merged to the main branch. A possible name could be *Fix Merged*. This could be useful, e.g., if additional testing is needed, for example by the submitter.
> - When there is nothing left to do for the issue, it can be *Closed*.
>
> In particular if we want to use Redmine to track all development and not just bugs (which I think would make the project status much more transparent for people outside the core team, and even within), these things should be considered to get the most out of Redmine.

All looks fine to me. More expressive option choices should lead to better group understanding of what is going on.

#8 - 06/30/2012 06:04 AM - Teemu Murtola

The names "Fix Submitted" and "Fix Merged" may not be the best possible, as the terminology doesn't apply that well to tasks or features. Don't have any better idea at the moment, replacing "Fix" with "Implementation" makes the state names quite long.

I agree that if Redmine always addresses the automatic fixing/closing to Anonymous, it isn't very useful. If it would support mapping the author name in the commit to a Redmine user, it would be more useful... And I also agree that it should not close the issue completely, but instead use another

state. The "Fix Merged" state (or whatever we want to call it) from my previous list should be the appropriate state for this.

#9 - 12/29/2012 07:34 PM - Erik Lindahl

Hi,

Trying to fix remaining things for release-4.6, I've experienced two concrete problems:

1) I'm not sure it's such a great idea to use redmine as the kitchen sink for *everything* that might or might not be related to new features, ideas, random users posting things that should rather to go the mailing list, etc. This adds up to quite a bit of noise that makes it harder to see the important bugs we need to fix.

2) When fixing things, you really want a simple list of things that still are not fixed, rather than having lots of "open" bug reports where need to open them and read individual commit messages to check whether they have been fixed, or worse: having to check gerrit whether anybody else has already fixed it. This is largely reflected in Teemu's comments above.

I think it would be good to use "resolved" to mark that the bug no longer needs work. This does not necessarily mean there is a patch in Gerrit. There was for instance bug where Berk mentioned he had solved it in his verlet tree, and it would be merged into release-4.6 in a couple of weeks. The typical action will likely be to have a brief explanation why it is resolved, so I think we could do without multiple extra states to describe exactly in what part of the Gerrit process the fix is - hopefully it is not going to stay long in that state anyway. Some bugs will also relate to documentation going e.g. on the web, so there too it will be a bit strange with the submitted nomenclature.

I also agree that it would be good with a "confirmed" state to help focus on things that really are bugs.

If we want a separate state to denote that a fix has been merged (but we want to close issues manually), I think "merged" is a good name for that :-)

#10 - 12/29/2012 07:49 PM - Szilárd Páll

Erik Lindahl wrote:

> 1) I'm not sure it's such a great idea to use redmine as the kitchen sink for *everything* that might or might not be related to new features, ideas, random users posting things that should rather to go the mailing list, etc. This adds up to quite a bit of noise that makes it harder to see the important bugs we need to fix.

I see your point, the noise-factor can indeed be a bottleneck. However, filters work pretty well, so if we decide that task/feature and all bug management should stay in redmine, we should try to use appropriate statuses and priorities (e.g. a minor unconfirmed bug should have low priority and open state). However, if we want to keep redmine more noise-free, we need some other platform for task/feature management.

> 2) When fixing things, you really want a simple list of things that still are not fixed, rather than having lots of "open" bug reports where need to open them and read individual commit messages to check whether they have been fixed, or worse: having to check gerrit whether anybody else has already fixed it. This is largely reflected in Teemu's comments above.

Adding more states and using priorities should help.

> I think it would be good to use "resolved" to mark that the bug no longer needs work. This does not necessarily mean there is a patch in Gerrit. There was for instance bug where Berk mentioned he had solved it in his verlet tree, and it would be merged into release-4.6 in a couple of weeks. The typical action will likely be to have a brief explanation why it is resolved, so I think we could do without multiple extra states to describe exactly in what part of the Gerrit process the fix is - hopefully it is not going to stay long in that state anyway. Some bugs will also relate to documentation going e.g. on the web, so there too it will be a bit strange with the submitted nomenclature.
>
> I also agree that it would be good with a "confirmed" state to help focus on things that really are bugs.
>
> If we want a separate state to denote that a fix has been merged (but we want to close issues manually), I think "merged" is a good name for that :-)

I agree that it would be nice, but considering the fact that this could be easily forgotten (and it's hard to enforce), I would not make this as a required step in the issue-management workflow. Having the state itself would be useful to get feedback on a gerrit change while in review. The ideal thing would be to have gerrit change the state and post a link to the change, but that will require a gerrit plugin which is probably not available (yet).

#11 - 12/29/2012 08:00 PM - Erik Lindahl

After exchanging another mail with Szilard, what about this suggestion for valid issue statuses (slight update of Teemu's list):

1. new
2. confirmed
3. rejected
4. feedback (this can be used after the bug is resolved/merged too - it simply means we want either info or confirmation of the fix from the submitter)
5. resolved
6. merged
7. closed

When we push a fix in Gerrit, we can simply set the status to resolved and paste a link to the Gerrit commit in the comment.

Any issue with a "fixes" keyword in Gerrit will be moved to the merged state automatically when merged into git, but the "closed" state is set manually.

#12 - 12/29/2012 08:42 PM - Szilárd Páll

I don't see the advantage of having a "fixes" keyword trigger the auto-setting of "merged" state, that's more or less obvious from the automatically linked "associated revisions". As a git commit that claims to fix a bug should do exactly that, I think it's fair to assume that this commit "resolves" the redmine issue.

Consequently, I suggest having "fix being tested" or "fix submitted" state for fixes in gerrit under review. However, unless we restrict the use of "feedback" status to the case when the fix has already been merged, we could use this for signaling that feedback is welcome on the fix under review. However, for this a developer would have to post a link to gerrit link.

#13 - 12/29/2012 08:51 PM - Szilárd Páll

Szilárd Páll wrote:

> Consequently, I suggest having "fix being tested" or "fix submitted" state for fixes in gerrit under review. However, unless we restrict the use of "feedback" status to the case when the fix has already been merged, we could use this for signaling that feedback is welcome on the fix under review. However, for this a developer would have to post a link to gerrit link.

On a second thought, this is not the best idea as the status would change in the following way:
new -> confirmed -> feedback -> resolved -> *feedback* -> closed
In this case the status would require to be changed manually to feedback after pushing to gerrit and again after merge which is IMO confusing.

Instead, it would be better to have:
new -> confirmed -> fix submitted -> resolved (auto) -> feedback -> closed
In this case feedback still needs to be picked manually, but the workflow seems more clear to me.

#14 - 12/29/2012 09:56 PM - Teemu Murtola

I would strongly advocate using Redmine also for task/feature management, and also for storing at least most important points about discussion related to them. As Szilard says, as long as the set of states and types of issues is well designed, it should be easy to create filters to focus on a particular set of issues. And we should create quick links to issue queries that are most often needed. Of course, the set of states and the types of issues depend a bit on the types of use that we have in mind. But I would really prefer a system where Redmine is used constantly during development, and not only dug up from naftalin once a year or so just before a release.

Some points, in no particular order:

- There are certainly things that now and then end up in Redmine, but perhaps shouldn't. But if some continuous effort is put into monitoring "New" tasks and rejecting inappropriate ones, those should not pile up such that they would harm normal use.
- I would strongly suggest to keep the "In Progress" state. Otherwise, there is no way to tell or communicate to others whether a "Confirmed" bug/feature is actually being worked on (or to find such issues using queries). Having such a state does not mean that we should force everyone to use it. But I have found it very useful when I have used Redmine as a todo-list for 5.0.
- In addition to the "In Progress" state mentioned above, I think the only real difference between Erik's list and mine is that I have two feedback states: one before the feature is ready ("Blocked"), and one afterwards ("Feedback"). This would simplify certain types of queries a lot.
- An alternative for a feedback-needed-for-in-progress-work ("Blocked") could be a separate "Discussion" issue type that could be created as a child "issue". Could be too complex a workflow, though.
- The reason why I would prefer to have at least part of the discussion related to new features etc. in Redmine is that in my experience, our mailing list discussions rarely reach a conclusion, and if a mailing list thread has been dead for a few weeks, no one is likely to resurrect it (or find it when work on the new feature actually starts, possibly months or even years later). In contrast, if at least the most relevant stuff is summarised in Redmine, it stays there, easy to find, and waits until someone has time to pick it up again.
- An alternative to separate "fix submitted" state could be a separate "Gerrit URL" field (or using the existing URL field for this purpose). This would allow one to also put, e.g., a link to an RFC change into an issue. Because there can be cases where it is nice to submit something to Gerrit for comments before the whole issue is resolved. A separate field for this purpose would make it easy to find the link without scanning all the comments. Need to check that we can give sufficient rights to users to do this, though.
- I think Redmine supports different sets of states for different types of issues, but we probably want to keep things simple.
- To me, "Resolved" and/or "Merged" feel better (more general) names than those that I originally listed.
- I think the original idea of "Feedback" in Redmine is that issues move from "Resolved" to "Feedback" *after* someone has provided some feedback. I think this is a better approach than having "Feedback" mean "feedback requested", because, again, this allows better queries for tasks that need some action. I understand that this conflicts a bit with my suggested use of the "Blocked" state, so this could require clarification.

I think the best approach is to spend a bit of time gathering input from different people, and then compose a concrete proposal for changing the states. This should be done carefully rather than fast, since if done right, it can help us a lot in the future.

#15 - 12/30/2012 11:56 AM - Erik Lindahl

I'm closer to Teemu's proposed naming here, since issues can be resolved on more ways than a specific fix having been submitted. Same thing when it comes to feedback: Normally users don't provide any feedback in 95% of the cases, even when we fix a specific bug for them, so assuming that will happen as the normal flow of every bug will just leads to dozens of issues piling up in the "feedback" state until we manually close them.

Realistically, not every issue is going to flow perfectly in a sequence through predetermined states since they might not fit in that model. Rather than trying to squeeze everything into one model, let's just provide a fair number of states that help describe in what part of the process the issue is. For Gerrit URLs, I think it's fine to just have them in comments in the discussion, otherwise we directly end up with problems if two different users submit different RFCs (and it won't require any field modifications).

The naming of the first blocked state isn't really critical. We know what it means, and when it comes to users there are really only two alternatives: Either they read the comments people make about their report or they don't, and in the latter case no naming in the world is going to help. Then we'd have something like

1. new
2. confirmed
3. in progress
4. rejected
5. blocked
6. resolved
7. merged (auto when things merge with a ref)
8. feedback
9. closed

This is pretty much the maximum amount of states I'd like to see, so if anything we should remove some, not add more.

#16 - 12/30/2012 01:17 PM - Teemu Murtola

I agree that we don't definitely want more states than those Erik listed. And on the other hand, that list is probably still manageable. I would only change "Confirmed" to "Accepted", since that is more generally applicable to tasks/features; "Confirmed" is quite specific to bugs. And I agree that we should not be too prescriptive in what transitions we allow. At least core developers should probably be allowed to do nearly any state transition. For anonymous/newly registered users, we probably only want to allow a few (like from resolved/merged to feedback). Don't know the details of the Redmine permission system, so can't comment in detail.

After we agree on the states (should we wait for comments from more people?), introducing the new states should get us started. We should also make a very brief guide to what the states mean and how to use them. Would the wiki be a good place for this? We can then adapt on the go if we see that something does not work. This is probably one other lesson to take from agile/Scrum (see my comment about Scrum in general in #949); try to continuously improve the processes as well, not only the code. I'm quite happy with the proposed set of states, so I have only one additional comment related to this aspect: if/when we reintroduce the "Resolved" state, it would be great if we could put it back such that it works as if it was never removed. Because I think that when it was removed, it was physically removed from the system, breaking all the history entries where an issue had been in a Resolved state. This may require hacking the backend database a bit, though, so perhaps it isn't a must thing to do.

#17 - 12/30/2012 08:45 PM - Roland Schulz

Overall LGTM. I'm not sure "feedback" defined as

> - If a user wants to provide feedback (e.g. confirming the bug has been fixed, or suggesting an even better fix), that is done by adding comments and selecting the 'feedback' state.

is good. For one I don't see why given feedback benefits from having its own status. For the given examples a comment without a status change is as good (given that statuses are IMO mostly useful for searching). Also it is confusing because the name "feedback" is currently used as meaning "feeedback required" not "feedback given". I would either remove the status "feedback" or keep its current meaning. That would let us distinguish between issues which are blocked by other issues and are blocked by required feedback.

#18 - 12/30/2012 09:29 PM - Teemu Murtola

Roland Schulz wrote:

> For one I don't see why given feedback benefits from having its own status. For the given examples a comment without a status change is as good (given that statuses are IMO mostly useful for searching).

I think that it is useful, because then there is a clear distinction between "resolved/merged" and "feedback". The first means that the implementer considers the issue done, and any action (other than possibly closing the issue) is the responsibility of someone else. "feedback", on the other hand, generally means that someone has made a comment that the implementer is responsible to react to. So it is very relevant to be able to distinguish these two states easily.

> Also it is confusing because the name "feedback" is currently used as meaning "feeedback required" not "feedback given". I would either remove the status "feedback" or keep its current meaning. That would let us distinguish between issues which are blocked by other issues and are blocked by required feedback.

Currently, "feedback" is the only state between "in progress" and "closed", so it is necessarily used for a mass of different things in an ad hoc manner. I think that the default Redmine workflow is that "feedback" follows "resolved" exactly in the meaning proposed here (so "resolved" means "feedback required/preferred", and "feedback" is "feedback given"). I don't think that the current, ad hoc usage is sufficient justification for not making it behave better.

For the "blocked" state, I agree that it can potentially mean a lot of things. But perhaps we can initially go with only one state there, and if we start piling up "blocked" issues that take too much time to shift through to find those that require some particular action, we then reconsider how to improve the situation.

#19 - 01/11/2013 05:30 PM - Rossen Apostolov

Since there weren't other comments I've added the new statuses. I changed 'confirmed' to 'accepted' as per Teemu's suggestion.

Should 'merged' correspond to an open state or to a closed state? At the moment only 'Closed' and 'Rejected' close the issues.

#20 - 01/11/2013 06:48 PM - Szilárd Páll

Teemu Murtola wrote:

> After we agree on the states (should we wait for comments from more people?), introducing the new states should get us started. We should also make a very brief guide to what the states mean and how to use them. Would the wiki be a good place for this? We can then adapt on the go if we see that something does not work. This is probably one other lesson to take from agile/Scrum (see my comment about Scrum in general in #949); try to continuously improve the processes as well, not only the code. I'm quite happy with the proposed set of states, so I have only one additional comment related to this aspect: if/when we reintroduce the "Resolved" state, it would be great if we could put it back such that it works as if it was never removed. Because I think that when it was removed, it was physically removed from the system, breaking all the history entries where an issue had been in a Resolved state. This may require hacking the backend database a bit, though, so perhaps it isn't a must thing to do.

I agree both with the need for documentation, which is in fact as important as adding the states, as well as in fixing the database properly. Unfortunately, like with the reported version field, the rushed removal of " Resolved" now requires it to be restored manually, but in my opinion this is *needed and not optional*.

#21 - 01/11/2013 07:01 PM - Szilárd Páll

Teemu Murtola wrote:

> Roland Schulz wrote:
>
> > For one I don't see why given feedback benefits from having its own status. For the given examples a comment without a status change is as good (given that statuses are IMO mostly useful for searching).
>
> I think that it is useful, because then there is a clear distinction between "resolved/merged" and "feedback". The first means that the implementer considers the issue done, and any action (other than possibly closing the issue) is the responsibility of someone else. "feedback", on the other hand, generally means that someone has made a comment that the implementer is responsible to react to. So it is very relevant to be able to distinguish these two states easily.

One could argue the other way around as well. If a user reports a bug, he/she probably does not care much about the chatter on the report page, but if nothing else "Feedback" (required), "Rejected", and "Resolved" would be of interest. Ideally, the "Feedback" message sent out as a notification to the reporter would be a quick and easy way for the developer(s) to ask for feedback. Otherwise this status would have no other use but to aid filtering. However, I would argue that as soon as feedback is requested, the developer working on the issue, typically being notified by mail, should got back to the issue, check the feedback and progress the issue to a further state if this is sufficient or leave it in "Feedback" state if it is not.

> Also it is confusing because the name "feedback" is currently used as meaning "feeedback required" not "feedback given". I would either remove the status "feedback" or keep its current meaning. That would let us distinguish between issues which are blocked by other issues and are blocked by required feedback.
>
> Currently, "feedback" is the only state between "in progress" and "closed", so it is necessarily used for a mass of different things in an ad hoc manner. I think that the default Redmine workflow is that "feedback" follows "resolved" exactly in the meaning proposed here (so "resolved" means "feedback required/preferred", and "feedback" is "feedback given"). I don't think that the current, ad hoc usage is sufficient justification for not making it behave better.

To me it sounds that what you mention as default redmine workflow is reasonable for the aforementioned reasons: if an issue was given feedback the watchers will anyway get notified and can/should take action. Of course, if the watchers don't care, many issues can get stuck in this state even after feedback is given, but I would rather not aim for dealing with ignorance by tailoring states to search for ignored issues.

#22 - 01/11/2013 07:13 PM - Szilárd Páll

Rossen Apostolov wrote:

> Since there weren't other comments I've added the new statuses. I changed 'confirmed' to 'accepted' as per Teemu's suggestion.
>
> Should 'merged' correspond to an open state or to a closed state? At the moment only 'Closed' and 'Rejected' close the issues.

I still don't see the point in having both "merged" and "resolved". If a bugfix or feature implementation gets merged it means that it passed code review and (as it should have the issue id in the commit message) reviewers should be familiar with the requirements by the issue and therefore only approve if the change *does resolve* the issue. Therefore, to me it seems logical that _merged = resolved _ - unless the code review missed some aspects, but there isn't much point in keeping an extra status and an extra manual change needed.

Additionally, for issues that don't get fixed through gerrit (the number of which should ideally be small), "resolved" is probably more appropriate.

So to conclude, unless somebody can point out a common use-case when both "merged" and "resolved" is useful to have, I vote for having only the latter.

#23 - 01/11/2013 08:07 PM - Teemu Murtola

- "merged" should be an open state. In particular if/when Redmine will automatically put issues into this state, we don't want them to get automatically closed.
- My idea of the "merged"/"resolved" difference would be that for an issue assigned to me, as soon as I have done the work (i.e., submitted it to Gerrit), I could change the issue to "resolved" and add the link to the Gerrit change. And when the change gets merged, Redmine automatically puts the issue into the "merged" state. This way, I can, at a quick glance of all the issues assigned to me, see where I can/need to do something in Redmine. It's not uncommon to have the review take so long that one can actually do some other work in-between. The same reasoning applies also for "feedback" (see also below).
- The submitter may not indeed be that much interested in the discussion going on between the "accepted" and "resolved" states (unless some feedback is required mid-way), but at least I, as a submitter, would be interested in knowing when the issue is resolved, whether it requires my feedback or not. So I would check the issues when they go to "resolved"/"merged" state and give any feedback if necessary. In general, we probably have more submitters than we have developers, and one submitter generally has less issues than one developer, so they have more time available for doing this. If an issue has no watchers and is not assigned to anyone (we have a lot of those...), or if the developer happens to be busy (not so uncommon, I would guess), I think it is very useful to be able to find those issues that need a comment without keeping the e-mails from Redmine as reminders and without keeping a separate TODO list.
- The idea behind all my proposals is to make it as easy as possible for the *developers* to find what they need to do, since (hopefully) they are the ones using the system the most. But this is of course important only if we actually want to use Redmine as a TODO list for the development. I have been partially doing that in the past two years for the analysis framework development, and these are just ideas coming out of that.

#24 - 01/11/2013 09:01 PM - Szilárd Páll

Teemu Murtola wrote:

> - "merged" should be an open state. In particular if/when Redmine will automatically put issues into this state, we don't want them to get automatically closed.
> - My idea of the "merged"/"resolved" difference would be that for an issue assigned to me, as soon as I have done the work (i.e., submitted it to Gerrit), I could change the issue to "resolved" and add the link to the Gerrit change. And when the change gets merged, Redmine automatically puts the issue into the "merged" state. This way, I can, at a quick glance of all the issues assigned to me, see where I can/need to do something in Redmine. It's not uncommon to have the review take so long that one can actually do some other work in-between. The same reasoning applies also for "feedback" (see also below).

I see your point, but I am personally still unconvinced. I think it's up to the review process and not the developer to decide if the issue is "resolved" and that's what the "Fixes #XYZ" should be for, i.e. if a merged commit contains such a reference, the review verifies the claim and approves that the commit indeed fixes #XYZ and therefore resolves it. Just take the example of a fix/implementation being rejected and the change set abandoned in gerrit. The developer will have to go back and switch *back* to "In progress" or whatever the state was before if we let the developer switch to "resolved".

The action taken when pushing up a fix/implementation to gerrit should be, if anything, switching to a "Patch submitted" or similar status + posting a link to the gerrit page. As I said before, this should ideally be done automatically, but as there is no plugin for it and ATM we don't have anyone free enough to implement it, we'd have to do it manually. With the above abandoned commit example, the worst case scenario is that an issue gets stuck in "Patch submitted" state after the patch is dropped (and the developer forgets to switch back). However, this is less of an issue than a patch getting stuck in "Resolved" state, right?

> - The submitter may not indeed be that much interested in the discussion going on between the "accepted" and "resolved" states (unless some feedback is required mid-way), but at least I, as a submitter, would be interested in knowing when the issue is resolved, whether it requires my feedback or not. So I would check the issues when they go to "resolved"/"merged" state and give any feedback if necessary. In general, we probably have more submitters than we have developers, and one submitter generally has less issues than one developer, so they have more time available for doing this. If an issue has no watchers and is not assigned to anyone (we have a lot of those...), or if the developer happens to be busy (not so uncommon, I would guess), I think it is very useful to be able to find those issues that need a comment without keeping the e-mails from Redmine as reminders and without keeping a separate TODO list.

I may be misunderstanding your reasoning, but as a submitter you could as well check for "Patch submitted" (which means that there is something to try out, but not necessarily final) or "Resolved" which IMO should mean that the issue is *considered resolved* by the developers, case in which even if the developers forget to set the status to "Feedback" the submitter can know and even search for that.

Additionally, from the developer's point of view I again don't see how does my proposal cause trouble. If the issue is accepted it really should be assigned, but even if it's not it is still clear the distinction between "Accepted" and "In progress" which should indicate that something is being worked on. Same goes for issues with a resolution under review, their status should, in my opinion, be "Patch submitted" (or equivalent) rather than "Resolved" which suggests to me that the developer him/herself is deciding whether the fix works/is suitable or not and this is not in line with our development process.

#25 - 01/12/2013 07:48 AM - Teemu Murtola

Szilárd Páll wrote:

> I see your point, but I am personally still unconvinced. I think it's up to the review process and not the developer to decide if the issue is "resolved" and that's what the "Fixes #XYZ" should be for, i.e. if a merged commit contains such a reference, the review verifies the claim and approves that the commit indeed fixes #XYZ and therefore resolves it. Just take the example of a fix/implementation being rejected and the change set abandoned in gerrit. The developer will have to go back and switch *back* to "In progress" or whatever the state was before if we let the developer switch to "resolved".

I don't want to start arguing about the semantics of the word "resolved". It can mean anything from "everyone is sure the issue is resolved" to "the person who implemented it thinks the proposed patch resolves it". As long as we agree that it is useful to have two states (and I don't see any argument in your reply disputing this), I don't care how they are called. And as long as the meaning is clear for everyone, the names should not matter that much. Feel free to change the names if you wish (just note that Gerrit has a "submit" button that does something very different from placing a patch in a "submitted" state in the meaning you propose here).

But you have a good point in that it could very well be better to have the latter state (i.e., the one that "Fixes #xyz" puts the issue in after getting merged) as the one that is shared with issues that don't require code changes. And for that use, "merged" is not a good name. Since the states Rossen added don't yet seem to be available for actual use, we are free to change them. And I propose that we finish this discussion before we make those states available.

> I may be misunderstanding your reasoning, but as a submitter you could as well check for "Patch submitted" (which means that there is something to try out, but not necessarily final) or "Resolved" which IMO should mean that the issue is *considered resolved* by the developers, case in which even if the developers forget to set the status to "Feedback" the submitter can know and even search for that.

I agree with what you say, but I don't see how it relates to the argument about "Feedback". My argument was that both meanings of "Feedback" don't make much difference for the submitter, but that it can make life easier to developers. To me, it seems that you are arguing exactly the same. I think that a very good way of requesting feedback for an issue is to assign it back to the submitter, no matter what state it is in (in most cases, this would happen in the "New" or "Resolved" states), so I don't see the need for a separate state for this purpose.

> Additionally, from the developer's point of view I again don't see how does my proposal cause trouble. If the issue is accepted it really should be assigned, but even if it's not it is still clear the distinction between "Accepted" and "In progress" which should indicate that something is being worked on.

I think we should not by default assign all "Accepted" issues to someone. To me, every "New" issue would ideally soon go to either "Accepted" or "Rejected", depending on whether it is a real bug/whether it matches the direction we want to take Gromacs to, no matter what is the schedule for possibly implementing it. In contrast, my first impression of "Assignee" is that that person is committed to working on the issue. And if every issue is already assigned to someone, it can discourage contributions. I think the barrier is lower for contributing to an unassigned issue, since it is much less likely that one would be duplicating work in such a case. But clear instructions here could help somewhat.

#26 - 01/13/2013 08:25 PM - Rossen Apostolov

- File workflow.jpg added

Teemu Murtola wrote:

> Since the states Rossen added don't yet seem to be available for actual use, we are free to change them. And I propose that we finish this discussion before we make those states available.

I discovered that the new statuses are not automatically enabled after addition. It is possible to enforce a workflow by allowing only a subset of new statuses depending on the current issue status. E.g. an issue that is "New" could be made into "Feedback" but not vice versa. The current defaults are http://redmine.gromacs.org/workflows/edit?role_id=3&tracker_id=1&used_statuses_only=0. I'm attaching a snapshot of what it looks like at the moment (the new statuses are not enabled

Do you prefer to allow conversion of all-to-all or enforce a workflow? At the very least it shouldn't be possible to change an issue to "New" once that status has changed.

#27 - 01/13/2013 08:37 PM - Szilárd Páll

Rossen Apostolov wrote:

> Do you prefer to allow conversion of all-to-all or enforce a workflow? At the very least it shouldn't be possible to change an issue to "New" once that status has changed.

Right now, without having discussed it with anyone, I think a workflow with strict and well-defined transitions is better than free for all and as you could see with other aspects of the issue management system, a complete freedom is not necessarily a good thing.

However, I strongly prefer that you **do not enable anything** yet. I prefer to **design** a workflow on "paper" first, present it to the developers, and switch only when we **know** and **are sure about** what we want and how do we want it. For instance, I would like us to create a flowchart of states, perhaps 2-3 for different types of issues and/or projects managed by Redmine, that would be the best way to both summarize the conclusions of this discussion including the alternatives if we are not converged to a single best naming scheme and workflow (which I'm not sure that we are just yet). Additionally a simple flowchart with a brief description would be the best as developer documentation which brings me to the next point. It is utterly)irresponsible* to change the working of the issue management system and not tell the developers what's changing and how to use it.

Therefore, I strongly oppose the introduction of any changes before the 4.6 final release as until then most developers will not even consider spending time reading through this lengthy discussion let alone actually giving it a thought.

#28 - 01/14/2013 01:37 AM - Szilárd Páll

Teemu, thanks for the detailed comments. I need to focus on getting some patches and other work done right now, so let me get back to you a bit later. I anyway think we should not rush this and should first make a decision, document it, and present the proposed workflow to the developers.

#29 - 03/07/2013 06:15 AM - Teemu Murtola

- File redmine-states.dot.pdf added

- File redmine-states.dot added

@Szilard: if you are serious about wanting a better workflow (with reference to discussion on gmx-dev), I propose that you start by continuing the discussion here... I have attached a proposed state diagram based on the discussion so far. This is essentially only a graphical representation of the e-mail that Erik sent a long time ago, with your feedback about resolved/merged addressed. If you are still unhappy with it, *please* provide *constructive* feedback on how it can be improved. I have left out several transitive state transitions that I think should be allowed to be not too prescriptive. I have not seen anyone comment that they would *not* want it this way (except by some minor comments here, for which my explanations have not received any reply). If we never do anything unless there is an unanimous consensus that everyone explicitly voices, we never get anywhere with the current level of activity... I'm tired of trying to beat the dead horse; please pick up the work from here if you want to do something extra.

#30 - 04/13/2013 11:39 AM - Teemu Murtola

Collected the proposal of how things should work here: http://www.gromacs.org/Developer_Zone/Redmine
Should cover stuff from here, #837, and Erik's e-mail.

If you think that something extra needs to be described in this "proposal", please be explicit in what you think is missing. Even better, contribute a suggestion.

#31 - 04/19/2013 04:22 PM - Mark Abraham

Rossen and I have done a partial implementation of the workflow of comment 29, following meeting with Erik, Berk and Szilard. We used some different wording. Feedback welcome, of course.

There's a Redmine version upgrade planned for the next week or two, so we'll see what new toys we will have available soon.

#32 - 04/19/2013 08:17 PM - Rossen Apostolov

- Status changed from New to Accepted

- Assignee set to Rossen Apostolov

#33 - 04/19/2013 08:17 PM - Rossen Apostolov

- Status changed from Accepted to In Progress

#34 - 04/19/2013 08:21 PM - Rossen Apostolov

Currently the setup enforces the setup described in the workflow diagram. One inconvenience is that one cannot bypass statuses, e.g. from New->Resolved directly. It needs to go New->Accepted->In progress->Resolved.

The good point though is that the workflow will be followed :)

Any preferences on that?

#35 - 04/19/2013 08:22 PM - Rossen Apostolov

- File workflow.jpg added

- Status changed from In Progress to Resolved

#36 - 04/19/2013 08:22 PM - Rossen Apostolov

Here's a figure with the current dependencies:

workflow.jpg

#37 - 04/25/2013 08:41 AM - Mark Abraham

We need a transition from Resolved to In Progress (see #1219) so that erroneous resolution decisions can be easily reverted.

I would suggest adding a transition from New to In Progress. We need Accepted as a holding state for things people think are worthwhile but which has not yet found someone with time to do. However, quick fixes don't need the extra state, and I think we want to make those as easy as possible to do. Likewise, New to Patch in Gerrit.

When we get a bot to start noticing new patches uploaded to gerrit, we will probably want more transitions to Patch in Gerrit, unless we can program the bot to follow multi-step transition paths.

#38 - 04/25/2013 08:44 AM - Mark Abraham

- Status changed from Resolved to Feedback wanted

Not resolved until we communicate at least a draft policy statement to gmx-developers. That should wait until Rossen has done the redmine upgrade

shortly and we know what the new toys can do.

#39 - 04/26/2013 06:21 AM - Teemu Murtola

I think it would be best to update http://www.gromacs.org/Developer_Zone/Redmine to match the final Redmine configuration, so that it is that policy statement. It should be relatively easy to update it, as it is mostly naming the states. The text needs to be updated to account for the different meaning of the "Feedback" state, though (which I still think would be better as "Feedback requires action", since the division of responsibilities between Resolved/Feedback/Closed would be clearer, but I'm not going to argue about this any further if people don't want to respond to my earlier comments). And perhaps add a link from the Redmine front page to that page, and include some how-to content there (like, what to fill in when submitting a bug report). We could also expand the text to include how to handle project versions, in particular keeping Target Version and the list of choices for it up-to-date.

Some minor issues:

- "Implemented / patch in gerrit" is a very long name for an issue state, and since Redmine does not seem to do wrapping for the Status column in queries, it will take a disproportionate amount of screen space if an issue in the list has this status.
- I agree that at least a few more transitions should be allowed:
  - Moving back from Resolved to In Progress and Implemented allows correcting for mistakes,
  - Moving from New to more states (even all states until Implemented) simplifies the workflow for simple issues, and similarly allowing skipping In Progress from Accepted.
  - Moving from Accepted to Blocked also removes an extra unintuitive state transition for cases where it isn't possible to start working on the issue because something is blocking. If this is added, Blocked to Accepted should also be added.
  - Moving from Implemented to Accepted is not in line with other transitions. Either it should be disallowed, or also moving from In Progress and Blocked to Accepted should be allowed.
- In the original Redmine configuration, people with a manager role in the project were allowed to do any state transition. It could be helpful if at least someone would be allowed to fix mistakes in the workflow. In the current configuration, only the New and Rejected states are unreachable from other states, though (and the Accepted state is unintuitive to reach).

#40 - 04/29/2013 07:58 PM - Teemu Murtola

Additional finding:

- Currently, marking issue as Resolved makes it disappear from queries for "open" issues. I think this is not appropriate, as the whole point of the workflow would be that Closed and Rejected are the only end states.

#41 - 04/29/2013 08:17 PM - Szilárd Páll

Teemu Murtola wrote:

> Some minor issues:
>
> - "Implemented / patch in gerrit" is a very long name for an issue state, and since Redmine does not seem to do wrapping for the Status column in queries, it will take a disproportionate amount of screen space if an issue in the list has this status.

I agree, the status name is too long. Everything goes through gerrit, so:

- there is no point in referring to gerrit explicitly;
- code uploaded to gerrit is called "change" and not "patch".

Hence, my not-so-creative suggestion would be "Change uploaded".

#42 - 04/29/2013 08:58 PM - Szilárd Páll

Teemu Murtola wrote:

> Additional finding:
>
> - Currently, marking issue as Resolved makes it disappear from queries for "open" issues. I think this is not appropriate, as the whole point of the workflow would be that Closed and Rejected are the only end states.

AFAIK that was intentional. I myself find it reasonable that there should be a state other than "Closed" or "Rejected" with the meaning that the bug has been fixed or the feature implemented and the code is thought to be doing what it claims to do. While I agree that in most cases the assigned developer should either request feedback or simply close the issue, I see "Resolved" as a state in between these two. In particular, switching the state automatically when a "fix" change gets merged to one which is considered "closed" should be the default behavior, I think.

On a second thought, allowing manual transition to the current "Resolved" (closed) is a catering bit for laziness, but IMHO it's better to use such a "Resolved" state than to leave a bunch of open issues assigned to past versions just because the assignee forgot to "clean up."

#43 - 04/29/2013 09:22 PM - Teemu Murtola

Szilárd Páll wrote:

> Teemu Murtola wrote:

> Additional finding:
>
> - Currently, marking issue as Resolved makes it disappear from queries for "open" issues. I think this is not appropriate, as the whole point of the workflow would be that Closed and Rejected are the only end states.

AFAIK that was intentional. I myself find it reasonable that there should be a state other than "Closed" or "Rejected" with the meaning that the bug has been fixed or the feature implemented and the code is thought to be doing what it claims to do. While I agree that in most cases the assigned developer should either request feedback or simply close the issue, I see "Resolved" as a state in between these two. In particular, switching the state automatically when a "fix" change gets merged to one which is considered "closed" should be the default behavior, I think.

If this is the case, I see no reason to have the "Closed" state. What possible purpose does it serve, if "Resolved" is in practise the state where every issue ends up and gets forgotten, and it is possible to go to "Closed" only from "Resolved"? Let's be honest: if the "Resolved" state does not show up in any queries as requiring some action, very few if any people are going to take a second look at it after the fix has been merged from gerrit. And the state of the issue can be anything when the change gets merged from gerrit, so this snapshot simply gets preserved.

My idea of the "Resolved" state would be that it is an open state, where it is still possible to check that the issue is really fixed and that information in the issue is accurate (e.g., the Target Version field). If we don't care about the closed issues at all (e.g., listing them in change logs or having the roadmap page in any way reflect reality), then it doesn't matter whether the information is accurate or not. However, closing issues that have been in the "Resolved" state for some time should be in most cases trivial (even if one checks that the data is correct).

> On a second thought, allowing manual transition to the current "Resolved" (closed) is a catering bit for laziness, but IMHO it's better to use such a "Resolved" state than to leave a bunch of open issues assigned to past versions just because the assignee forgot to "clean up."

If such a manual transition would not be allowed, then it would be *impossible* to have issues that don't get resolved through gerrit (e.g., it would be impossible to close this issue).

#44 - 05/20/2013 02:45 PM - Rossen Apostolov

- File wf.jpg added

#45 - 05/20/2013 02:47 PM - Rossen Apostolov

I tried to address the comments regarding the workflow. Here is how it looks at the moment:

wf.jpg
I renamed the "implemented in gerrit" to "fix uploaded".

Do the current settings correspond to what you see as a workflow?

#46 - 05/20/2013 05:20 PM - Mark Abraham

In discussion this afternoon, we all want the plugin that changes the state to "fix uploaded." That transition thus needs to be available from every starting point, since we cannot require all developers to jump through bureaucratic hoops when they find a bug. For example, in #1255 it is great that Michael opened a Redmine issue for a short-lived bug fix, but it is silly for the system to require him/me/someone to move the status manually through several stages for such a short-lived issue. Also problematic are any one-way transitions that might be selected through someone's error.

For user-reported bugs awaiting triage, or accepted bugs waiting for someone to choose to spend time to fix them, or when feedback is needed, the detailed statuses are useful. We can only invite people to use them, and accordingly, I think the barrier to their use should be as low as possible.

So, we think it is best to document the intended use of the statuses, install this plugin, publish an (updated) status workflow guideline on the wiki (at least for now), allow an all-to-all status transition matrix, document the current git-redmine-gerrit-jenkins workflow that we want people to use, publish that on gmx-developers, and invite further feedback.

#47 - 05/21/2013 02:16 AM - Szilárd Páll

Rossen Apostolov wrote:

> I renamed the "implemented in gerrit" to "fix uploaded".

As discussed before, besides bugs we have tasks and features in redmine and for these "fix" is not the right wording.

#48 - 05/22/2013 10:07 AM - Mark Abraham

Szilárd Páll wrote:

> Rossen Apostolov wrote:
>
> > I renamed the "implemented in gerrit" to "fix uploaded".
>
> As discussed before, besides bugs we have tasks and features in redmine and for these "fix" is not the right wording.

We can live with that. I don't think there is a perfect wording that is clearly appropriate both for the putative fixes that are patches uploaded to gerrit, and to fixes of things that are not going via gerrit.

One could have two different statuses and disable them for different projects, but I don't see the added value paying for the maintenance.

#49 - 05/22/2013 10:10 AM - Rossen Apostolov

Right, and actually, "fix" was chosen to refer to "fixing the filed issue in redmine", not the type of tracker that it represents (bug, feature, etc.).

#50 - 05/23/2013 05:47 AM - Teemu Murtola

I don't think the wording matters that much, although I agree that "fix a feature" is not the right word. I originally proposed "Implemented", which I guess (but can't know, since no one has bothered to justify the changes relative to the original proposal) is not unambiguous in whether the implementation is already part of the main branch. But that should work for all types of issues.

I would also consider renaming the "Feedback wanted" to just "Feedback". Since my original proposal for the semantics seems to be controversial, I propose a compromise: this state would mean that *some* action is needed related to the feedback. Either feedback is requested, or feedback that requires action has been provided. To me, it seems that it is anyways used in this way already now...

#51 - 05/24/2013 10:46 AM - Rossen Apostolov

Mark Abraham wrote:

> In discussion this afternoon, we all want the plugin that changes the state to "fix uploaded."

Found that: https://github.com/tru/redmine-gerrit-scripts, it should do what we want. It's rather old and not tested with latest redmine, though.

#52 - 05/24/2013 11:31 AM - Teemu Murtola

Rossen Apostolov wrote:

> Mark Abraham wrote:
>
> > In discussion this afternoon, we all want the plugin that changes the state to "fix uploaded."
>
> Found that: https://github.com/tru/redmine-gerrit-scripts, it should do what we want. It's rather old and not tested with latest redmine, though.

It won't do without heavy modification. It has several issues in handling either multiple issue references in one commit, or cases where multiple commits are necessary to resolve an issue, or links like "Related to #xyz". A separate post for each patchset may also be excessive.

#53 - 05/24/2013 02:22 PM - Rossen Apostolov

Here's the todo list from Mark's comment:

1. document the intended use of the statuses
2. install this plugin (tracking in #1264)
3. publish an (updated) status workflow guideline on the wiki (at least for now)
4. allow an all-to-all status transition matrix (DONE)
5. document the current git-redmine-gerrit-jenkins workflow that we want people to use
6. publish that on gmx-developers
7. invite further feedback

#54 - 05/31/2017 05:41 PM - Szilárd Páll

Can you please update what's the status of the 7 TODOs, what's new since those decisions made 4 years ago (e.g. new redmine features)?

Files

workflow.jpg	125 KB	01/13/2013	Rossen Apostolov
redmine-states.dot.pdf	20 KB	03/07/2013	Teemu Murtola
redmine-states.dot	958 Bytes	03/07/2013	Teemu Murtola
workflow.jpg	71 KB	04/19/2013	Rossen Apostolov
wf.jpg	57.1 KB	05/20/2013	Rossen Apostolov