Redmine - Feature #2182

Weighted version completion percentage

2008-11-13 21:09 - Bobby Birks

Status:	Closed	**Start date:**	2008-11-13
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Roadmap	**Estimated time:**	0.00 hour
Target version:	0.9.0		
Resolution:	Fixed		

Description

Showing percentage of issue completion is a handy feature, but it would be nice to have the ability to see an estimation of work completion. It could be more intuitive in some cases. I would like to suggest this feature as at least an **option** to replace existing functionality, whether as an program, project, target version, or user-level preference. If so enabled, it should replace the current means of calculating version completeness percentages (and showing progress bars) wherever they appear.

As an example, a scattering of estimated times on Version Foo of a project:

Issue	Estimated	Status	Percent Done	Time Spent
1	16	Open	0%	0.0
2	4	Closed	100%	4.0
3	3	Open	40%	1.3
4	Unknown	Open	0%	0.0
5	Unknown	Open	50%	3.5
Total	23			8.8

As I understand it, the current methodology would consider each issue equally. So a 32 hour issue that's 50% done has equal value to a 0.5 hour issue that's 50% done. So in our case, we get

1. 0% / 5 = 0%
2. 100% / 5 = 20%
3. 40% / 5 = 8%
4. 0% / 5 = 0%
5. 50% / 5 = 10%

This totals 38% done.

Unfortunately, I cannot think of any alternative that doesn't involve estimated time in some fashion, and that is an optional field.

While Redmine already displays total estimated/spent time for a version, using this directly could be flawed for several reasons. These reasons include that tracking time spent is not required either, and that there's nothing preventing spent time from exceeding estimated time.

Perhaps the most compatible option would be to split a version's issues into estimated and non-estimated. In the above example, that would be 3/5 estimated and 2/5 non-estimated issues.

The non-estimated issues would be given the same weight all issues are currently given. In this case, 20% (or one fifth).

The estimated issues are weighted relative to each other. There may be a more mathematically efficient representation, but below is a general idea:

1. We have a total of 23 estimated hours for issues that were estimated.
    1. Weighted 16/23
    2. Weighted 4/23
    3. Weighted 3/23
2. 60% of the versions issues have been estimated, so we must adjust the weights accordingly
    1. Weighted 48/115
    2. Weighted 12/115

    3. Weighted 9/115
3. Now we consider the completion of the issue with regard to its weight
    1. 0%, no need to apply the weight
    2. Closed, no need to apply the weight
    3. 40% done, weighted by 9/115 gives 18/575, or about 3% completion of the overall project (instead of 8%)

In this example, the difference would be relatively small (33% instead of 38%). However, we have a target version in our Redmine installation which shows 38% completion, but by the above logic it would be closer to 10%. In our case it's due to fixing a large number of small, quick (1 hour or less) bugs before starting on big new features (ranging from 4-8 estimated hours on average).

Related issues:

Related to Redmine - Feature #2139: Roadmap: use estimated time for calculati...	**Closed**	**2008-11-06**
Related to Redmine - Defect #4682: Completed version with wrong progress bar ...	**Closed**	**2010-01-28**

Associated revisions

Revision 2349 - 2009-02-01 19:54 - Jean-Philippe Lang

Use estimated hours to weight issues in version completion calculation (#2182).

History

#1 - 2008-11-13 22:25 - Eric Davis

This is similar to #2139. I've customized my Redmine to take advantage of the estimated time. The formula I use is:

```
All issues are filtered to the specific Version

OITE = Open Issue Time Estimate
OIPC = Open Issue percent complete (e.g. 50, 60, 100)
CITE = Closed Issue Time Estimate
ATE = All issue time Estimates

% complete for Version A = (Sum(OITE * OIPC) + Sum(CITE * 100)) / Sum(ATE)
```

This equation will not factor in issues without a time estimate at all. Then for the case when a version's issues have no time estimate, I fall-back to the default way of calculating percent complete based on number of issues.

#2 - 2008-11-13 22:40 - Carl Nygard

I've seen this issue crop up on the discussion boards and it's always bothered me. I just now figured out why.

To me this seems to be a lot of effort and especially complication for something that still doesn't give you complete information. If you're looking at the roadmap, and you see 10% vs. 38% (from your example), you still do not know **of what**. 4hr? 40 days? 3 wks? It's still useless, even if you think it's more accurate.

What you're really looking for is Time-to-completion, which would be a tally of all outstanding work to be done on the issues related to the project. Simple SUM (estimatedTime * [100-percentComplete]) or perhaps SUM (estimatedTime-actualTime) over all open issues.

To be complete, you'd probably want a count of the issues that don't have estimates, as well as a count of issues that have overrun the estimates with actual time. In fact, I'm sure a few more metrics could be applied (can anyone say plugin?) such as count of issues overrun, total time overrun, etc.

This would give you much more specific information about estimates, actuals, and unknowns, instead of burying it all inside a single suspect percentage-complete calculation.

#3 - 2008-11-13 23:50 - Bobby Birks

Sorry, I missed that issue. All variations I searched on involved "percent" to catch percent and percentage :(

Fortunately, I was aware of time to completion being suggested suggested in #1953 (but I thought this issue might be quicker implement due to smaller scope, and technically is separate from that).

After reading #2139 and reviewing Eric's formula, I wonder what would be better -- falling back to the old method if even one issue lacks an estimate, or assuming it's estimate is going to be "average." I can see arguments both ways. Particularly, my suggestion is less intuitive to someone who does not know the processes involved.

My understanding is that the proposed options are:

- Make estimated time required (which probably won't work for a lot of people, but putting it out for sake of completeness)
- Falling back to the current method if there's even one missing an estimated time
- Completely skipping issues lacking an estimated time (probably falling back to the current method if they're all that way)
- Treat the non-estimated issues as average (which should be the net effect of my suggestion) and fall back to the current method if they're all that way

#4 - 2008-11-14 02:15 - Toni Kerschbaum

Carl Nygard wrote:

> I've seen this issue crop up on the discussion boards and it's always bothered me. I just now figured out why.
>
> To me this seems to be a lot of effort and especially complication for something that still doesn't give you complete information. If you're looking at the roadmap, and you see 10% vs. 38% (from your example), you still do not know **of what**. 4hr? 40 days? 3 wks? It's still useless, even if you think it's more accurate.

I think that while having a Time-to-completion feature would be very useful too, the proposed solution(s) aren't useless because the total estimated time for every version is calculated and displayed after clicking on said version.

The ansatz posted by Bobby seems very sophisticated but not hard to implement to me and would probably be the best way for calculating the progress of a version. Maybe I'll try it myself, but unfortunately I don't know Ruby. So, while I'm trying, this gets +1 by me ;)

Eric Hulser:
I don't know how much work it is to make a patch, but if it is not too much, could you maybe please post one for your implementation?

#5 - 2008-11-14 06:04 - Eric Davis

- File percent_from_hours.diff added

Carl Nygard wrote:

> To me this seems to be a lot of effort and especially complication for something that still doesn't give you complete information. If you're looking at the roadmap, and you see 10% vs. 38% (from your example), you still do not know **of what**. 4hr? 40 days? 3 wks? It's still useless, even if you think it's more accurate.

I agree, just a % doesn't help much unless you understand the formula behind it. I use the percent to track projects for my customers so I can say "we're half done so the ETA would be x".

Bobby Birks wrote:

> My understanding is that the proposed options are:
>
> - Make estimated time required (which probably won't work for a lot of people, but putting it out for sake of completeness)
> - Falling back to the current method if there's even one missing an estimated time
> - Completely skipping issues lacking an estimated time (probably falling back to the current method if they're all that way)
> - Treat the non-estimated issues as average (which should be the net effect of my suggestion) and fall back to the current method if they're all that way

Another option is treat all non-estimated issues as 1 (or another set number). This allows all issues to weigh into the version without making the estimated time field required.

> I don't know how much work it is to make a patch, but if it is not too much, could you maybe please post one for your implementation?

I've attached a crude patch. I've cleaned it up in another branch but I don't have access to it right now.

```
# app/models/version.rb
   def completed_pourcent
     if fixed_issues.count == 0
       0
     elsif open_issues_count == 0
       100
     else
-      (closed_issues_count * 100 + Issue.sum('done_ratio', :include => 'status', :conditions => [
"fixed_version_id = ? AND is_closed = ?", id, false]).to_f) / fixed_issues.count
+
#      (closed_issues_count * 100 + Issue.sum('done_ratio', :include => 'status', :conditions => ["fixed_versi
on_id = ? AND is_closed = ?", id, false]).to_f) / fixed_issues.count
+      calculate_percent_from_hours
     end
   end

# app/models/version.rb
+
+  # Calculates the % complete on a project based off the total esimated
+  # time left on the issues divided by the total time
+  def calculate_percent_from_hours
+    spent = 0
```

```
+    total = 0
+    Issue.find(:all, :include => 'status', :conditions => ["fixed_version_id = ? AND #{IssueStatus.table_name
}.is_closed = ?", id, false]).each do |i|
+      unless i.estimated_hours.nil?
+        spent += i.estimated_hours * i.done_ratio
+        total += i.estimated_hours
+      end
+    end
+
+    Issue.find(:all, :conditions => ["fixed_version_id = ? AND #{IssueStatus.table_name}.is_closed = ?", id,
true], :include => :status).each do |i|
+      unless i.estimated_hours.nil?
+        spent += i.estimated_hours * 100
+        total += i.estimated_hours
+      end
+    end
+
+    if spent == 0 and total == 0
+      # Redmine default
+      return (closed_issues_count * 100 + Issue.sum('done_ratio', :include => 'status', :conditions => [
"fixed_version_id = ? AND #{IssueStatus.table_name}.is_closed = ?", id, false]).to_f) / fixed_issues.count
+    else
+      return spent / total
+    end
+  end
```

#6 - 2008-11-14 18:32 - Jean-Philippe Lang

- File version_completion.patch added

I've made a patch for this.
It uses estimated time to weight each issue. Issues with no estimated time are weighted with the average estimated time.
If no issue are estimated, they receive the same weight, so we fall back in the current calculation. For this reason, I think we don't have to make this new calculation method an option.

Let me know what you think before I commit this patch.

#7 - 2008-11-14 18:48 - Jean-Philippe Lang

- File deleted (version_completion.patch)

#8 - 2008-11-14 18:48 - Jean-Philippe Lang

- File version_completion.patch added

#9 - 2008-12-13 02:30 - Toni Kerschbaum

I just wanted to check on the progress of this feature and found your patch :)
We'll test it asap (next Monday I think) and report back, but your idea sounds very nice to me.

Thanks for all your efforts, Eric and Jean!

#10 - 2008-12-13 19:49 - Toni Kerschbaum

- File perc.gif added

I already tested your patch and it seems to work great, except one small thing:
The small line under the percentage bar displays a false (old) progress, see attached image.

#11 - 2008-12-13 19:53 - Toni Kerschbaum

- File perc_bug_02.gif added

Ehm, sorry, I uploaded a bad picture, the problem isn't really understandable with it. Here is a better one.

#12 - 2008-12-18 08:18 - Eric Davis

Toni Kerschbaum wrote:

> I already tested your patch and it seems to work great, except one small thing:
> The small line under the percentage bar displays a false (old) progress, see attached image.

I'm not sure if I would consider that a bug or not. The text is talking about the number of open issue and the number of closed issues so I would think

a basic % would be best here. In Toni Kerschbaum's case, 1 open issue out of 4 issues is 25%.

#13 - 2008-12-19 22:23 - Toni Kerschbaum

Eric Davis wrote:

> Toni Kerschbaum wrote:
>
> > I already tested your patch and it seems to work great, except one small thing:
> > The small line under the percentage bar displays a false (old) progress, see attached image.
>
> I'm not sure if I would consider that a bug or not. The text is talking about the number of open issue and the number of closed issues so I would think a basic % would be best here. In Toni Kerschbaum's case, 1 open issue out of 4 issues is 25%.

You are right, if you look at this that way, it really makes sense and should be left that way.
Thanks again for your support!

#14 - 2009-02-01 19:50 - Jean-Philippe Lang

- Category set to Roadmap

- Status changed from New to Closed

- Target version set to 0.9.0

- Resolution set to Fixed

Patch committed with slight changes in r2349.

Files

percent_from_hours.diff	4.15 KB	2008-11-14	Eric Davis
version_completion.patch	1.6 KB	2008-11-14	Jean-Philippe Lang
perc.gif	949 Bytes	2008-12-13	Toni Kerschbaum
perc_bug_02.gif	816 Bytes	2008-12-13	Toni Kerschbaum