# Redmine - Feature #465

## Inheritence of Versions to Subprojects

2007-10-30 05:46 - Dirk Szameitat

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | Jean-Philippe Lang | **% Done:** | 0% |
| **Category:** | Projects | **Estimated time:** | 0.00 hour |
| **Target version:** | 0.9.0 | | |
| **Resolution:** | Fixed | | |

**Description**

It would be really nice, if issues of a subproject could be set to the fixed version defined in the parent project.
This should be made visible in the roadmap page as well o get an overview of all issues for each version without having to change to another project.

**Related issues:**

| | | |
|---|---|---|
| Related to Redmine - Feature #2666: Roadmap for main project should see Roadm... | **Closed** | **2009-02-04** |
| Related to Redmine - Feature #4932: Inheritence of Versions to Subprojects - ... | **Closed** | **2010-02-25** |
| Has duplicate Redmine - Feature #2063: Share version of project and sub-project | **Closed** | **2008-10-21** |
| Has duplicate Redmine - Patch #774: bulk edit - subprojects inherits versions... | **Closed** | **2008-03-03** |
| Has duplicate Redmine - Feature #4341: same version for child projects | **Closed** | **2009-12-04** |
| Has duplicate Redmine - Feature #3105: Roadmap and subrojects dependencies. | **Closed** | **2009-04-02** |
| Has duplicate Redmine - Feature #3430: Main project version management : link... | **Closed** | **2009-05-29** |
| Precedes Redmine - Feature #4357: Show shared versions on the Project Settings | **Closed** | **2009-12-07** |

### Associated revisions

**Revision 3123 - 2009-12-06 11:28 - Jean-Philippe Lang**

Version sharing (#465) + optional inclusion of subprojects in the roadmap view (#2666).

Each version of a project can be shared with:

- subprojects
- projects in the project hierarchy: ancestors + descendants (needs versions management permission on the root project)
- projects in the project tree: root project + all its descendants (same as above)
- all projects (can be set by admin users only)

Notes:

- when sharing a version of a private project with others projects, its name will be visible within the other projects
- a project with versions used by non descendant projects can not be archived

**Revision 3130 - 2009-12-07 20:28 - Jean-Philippe Lang**

Fixes Project#shared_versions for descendants sharing (#465).

### History

**#1 - 2008-01-02 18:15 - Mark Elrod**

+1

I was hoping that I could define my versions once for a parent
project and then have the sub-projects pick them up rather than
have to change each one for every new version we create.

**#2 - 2008-06-12 08:37 - Jani Tiainen**

+1

Specially since you already can inherit issue categories and user roles, why left versions out?

**#3 - 2008-06-12 10:51 - Paul Rivier**

I also think it would be natural.

There must be some reasons why it is not the case now, does anybody know them ?

#### #4 - 2008-06-18 18:34 - Paul Rivier

*- Assignee set to Jean-Philippe Lang*

Hello all,
I am now sure I do need versions inheritance. This might be because my use cases are more about project management (non software) than specific software, therefore I use versions as project milestones. Jean Philippe, I am assigning this issue to you, I would like to have your point of view on this. Please let me know whether :

- [ ] it is something NOT TO DO, and explain briefly why
- [ ] it is something desirable, but with special care in implementation
- [ ] no problem, I can do it straight and send patch
- [ ] ne se prononce pas

cheers

#### #5 - 2008-06-18 21:43 - Jean-Philippe Lang

It's something desirable indeed.

But I think we should provide a way to enable/disable this inheritance. Maybe one of the following option:

1. a global setting: "subprojects inherit versions"
2. a flag on each parent project to enable/disable versions inheritance on its subprojects
3. a flag on each subproject to enable/disable versions inheritance

Option 3. seems to be the most flexible. What do you think ?

#### #6 - 2008-06-18 21:46 - Jean-Philippe Lang

Also a 4th option: a flag on each parent version to enable/disable its inheritance on subprojects (even more fine grained).

#### #7 - 2008-06-18 22:36 - Paul Rivier

I think option 3 fulfill what most user needs.
I like option 1 and 2 because of their simplicity, although 1) is probably too coarse for some cases.

Some more thoughts, because we are note puzzled enough yet :)

- **Inferred inheritance** : Don't show in *SubProject B* roadmap versions from Master_Project A that *does NOT have any SubProject_B issue linked to it*. Only show Master Project A versions in the dropdown list from "New Issue" view. Can be combined with the 3 options you proposed above.
- **Versions linking** : we can, more generally, provide a way to 'link' (or 'import') versions from any project to an other. It would be really like hardlinking in the filesystem. That would probably require a change in the model, though.

Again, I think any of the options we are talking about will provide at least 99% of satisfaction for our needs, so no big deal here.

#### #8 - 2008-06-20 16:06 - Paul Rivier

Jean Philippe, can you tell me if you plan to implement this yourself ? I have started something, but because it is not so evident, and quiet at the core of redmine, you might prefer to do it yourself. Please let me know if I should do it myself or wait, thanks.

#### #9 - 2008-06-21 17:44 - Krzysztof Starzecki

I would be very happy to see this feature in 0.8

#### #10 - 2008-06-24 19:37 - Paul Rivier

I'm working on it. It currently works against 0.7.2. It is more work than what I thought first. :)
I'm having some difficulties at some levels, mainly regarding permissions.

- I can not find a way to check whether a user can view or not versions, and issues
  -- JP, any pointer ?

- I don't know what to do when :
  - Joe has registred "issue 3" to parent project "version A" (Joe has relevant rights in parent project)
  - Sam looks at "issue 3", but he has no right to see parent project
    -- Currently, target version will show and link "Version A", but of course clicking on it pops a 403.

What do you think I should do ?

#### #11 - 2008-06-24 20:36 - Jean-Philippe Lang

Concerning 1, user.allowed_to?(:view_issues, project) returns true if user is allowed view issues on project, otherwise false (there is no specific

permission to view issue, so if he can view issues then he can view versions).

Concerning 2, I think we can make it simple. If the parent project is private, then it's up to the admin not to make public subprojects inherit parent's versions. Maybe we could disable the link to the version (just display its name) if the user is not allowed to view it.

Btw, where do you store the flag to enable/disable inheritance ?

#### #12 - 2008-06-25 14:07 - Paul Rivier

Hi Jean-Philippe,

> Concerning 1, user.allowed_to?(:view_issues, project) returns true if user is allowed view issues on project, otherwise false (there is no specific permission to view issue, so if he can view issues then he can view versions).

Ok, that's what I did. So if one day, a :view_versions permission appears, we must remember to change the source.

> Concerning 2, I think we can make it simple. If the parent project is private, then it's up to the admin not to make public subprojects inherit parent's versions.

This is too coarse I think, because members of both projects would therefore be unable to share versions between them, while this seems to be a common scenario. Also, I consider version inheritance settings should be let to projects managers, not to redmine administrator. WDYT, please ?

> Maybe we could disable the link to the version (just display its name) if the user is not allowed to view it.

Because the name itself is a sensitive information (like "We buy company X"), my point of view is that even the name should be hidden.
What do you think ?

> Btw, where do you store the flag to enable/disable inheritance ?

So far nowhere because I have not changed the model. In other words, so far it is enabled by default and for all. But I filter the roadmap view so that empty versions from parent project do not show up, as proposed above. WDYT ?

Also, afer more thoughts, I think I would like to change the model to use the following flags :

- On version model : "Subprojects inherit this version" (true/false)
- On Project model : "This project inherit versions from parent projects" (true/false)

That way, one can easily set a subproject to inherit from parent, but can, on a specific version of parent project, prevent it from being iherited even in subprojects with version_inheritance flag set to :true. WDYT ?

During development, I try to keep in mind that one day redmine could ofter more depth in projects trees. Although current code wont work magically, it will be very easy to adapt.

#### #13 - 2008-06-27 13:55 - Paul Rivier

*- File veersion_inheritance.patch added*

## Important notes about versions inheritance patch

### Before reading or applying this patch, you should be aware of :

1. author is not an experienced rails developer (at all)
2. he is even less a web application developer (at all)
3. he only knows some ruby (and some LISP of course for the sake of his soul)
4. the patch has been developed against current stable 7.2.0
5. it does not alter the model, hence there is no setting to enable/disable version inheritance, so applying it activates automatically the feature for all projects
6. it won't work when Redmine has more than 2 levels of depth in the projects trees.

So you are warned :)

### What does the patch do ?

To make things clear lets take a simple use case :

- Project_B is a sub-project of Project_A
- Joe can view Project_B, but not Project_A
- Tom can view Project_A, but not Project_B
- Anna can view both Project_A and Project_B

Version inheritance in activated.

Anna:

- When reporting issue for Project_B, Anna can see Project_A versions, and attach issue to them,
- but when reporting issue for Project_A, Anna won't see Project_B versions.

Joe:

- When reporting issue for Project_B, Joe can NOT see Project_A versions
- If Anna has reported some issue in Project_B, attaching them to a project_A version, Joe won't see this information.

Tom:

- When viewing Project_A roadmap, Tom can't see Project_B issues attached to Project_A versions

**To sum-up, the patch activates version inheritance and try hard to keep private information safe**

## Why does this patch suck ?

Information privacy is NOT handled from a central point. This means that I had to go to every single place where I expect some sensitive information to appear, then I put some special switch to hide it.
I consider this is a very poor implementation, but I could not do better.

## Why I did not better ?

It might be because I'm not experienced at all with rails and with redmine. But also, I think there is something lacking in current design of redmine. There is no layer between database, and data provided to views. Most of the time, fine-grained permission checking is done right in the view. Of course there is this before_filter blocking rendering totally if action is not allowed, but once its allowed, the controller and the view will happily pick up data directly from the database, and apply some ad-hoc filtering rules whenever required. This is why I could not find a way to guarantee that a Joe will never see versions from Project_A. All I can do is going in every places where versions are about to be printed to ensure special filtering.

In other words, even when you are browsing as an anonymous, it is up to each action to do the correct database query, then filter out what you should not see.

I imagine there could be an intermediate security layer. It would take care of database queries. Above this layer, database queries are prohibited. So if you want collection of issues for project A, you just ask them to the filtering layer. This ensure there is a **single, unavoidable** passage in the application where data transfert from database to controllers/views occur. If I could find this place, I would only had to add 2 or 3 filtering rules to the issues provider, and I could claim that Joe will never see version from Project A.

Also, I don't know how other web applications do that. Any pointer to an application known to do that very well is welcome.

## Conclusion

I won't provide a patch for development branch because I'm unhappy with current code, so I can't wish to redmine 0.8 to integrate it in its current form. Also, I think some strong directive should emerge on how projects are related to each others, so that implementation can be generalized to arbitrary depth of nesting of projects. Jean Philippe, I hope this will happen soon.

For those needing absolutly this feature today, this patch is for you.

Finally, because I know very little about web development, I am probably wrong on many points, and the code could probably be better, so please feel free to comment.

#### #14 - 2008-07-01 12:55 - Paul Rivier

for what it's worth, I added the model described above. For the record, it

- adds an 'inheritable' property to versions
- adds on 'inherit_versions' property to projects

Both are booleans. If you want them, just add them to the model, and use the following function in project.rb

```
def related_versions(user=User.current)
   versions=self.versions
   if self.inherit_versions &&
       self.parent &&
       user.allowed_to?(:view_issues,
                        self.parent) then
     versions += Version.find(:all, :conditions => {:project_id => self.parent_id,
                                 :inheritable => true})
   end
   return (versions.sort{ |a,b|
              ad=a.effective_date
              bd=b.effective_date
```

```
              if ad == bd
                a.name <=> b.name
              elsif (not ad)
                1
              elsif (not bd)
                -1
              elsif ad > bd
                1
              elsif ad < bd
                -1
              else 0
              end })
  end
```

#### #15 - 2008-07-04 09:27 - Michael Härtig

Thank you for this patch. It works and it is that what I want. So hopefully this goes to 0.8

#### #16 - 2008-07-29 09:52 - Jan Ivar Beddari

Michael Härtig wrote:

> Thank you for this patch. It works and it is that what I want. So hopefully this goes to 0.8

Thanks to Paul for doing this patch and helping me learn more about Redmine. Keep up posting those long discussions, it helps me and others that are learning, just as you are! :-)

I have software engineering experience but not with Rails. I very much agree with you Paul that this really should NOT go into 0.8 because of the limitations in the current code base and design. It'd be very stupid to do too much too quickly. I'm actually a bit suprised that subprojects even exist in their current form ..

#### #17 - 2009-01-07 20:12 - Keith Harry

Are there any plans to merge this into an official release?

#### #18 - 2009-01-08 10:41 - Paul Rivier

There has been some development recently by jp_lang in the 'work' branche to remove current limitation in project nesting depth. With arbitrary depth, nesting projects will become a key feature to reflect organisation, and not just a convenience in the way project names and activity are displayed.
A new issue could be open to discuss how nested projects should be related to each others in terms of issues, activity, milestones (versions), wiki and so on.
Jean Philippe, do you have some ideas already, do you also think a lot of power can be derived from projects nesting ?

#### #19 - 2009-04-13 16:56 - Kevin Glowacz

Now that unlimited sub-projects have been merged into trunk, I'd love to see Inheritence of Versions to Subprojects

#### #20 - 2009-06-22 11:39 - Benjamin Baroukh

+1

#### #21 - 2009-06-22 14:32 - Brahim Abdesslam

+1

#### #22 - 2009-06-22 14:39 - Nanda P

+1

#### #23 - 2009-06-24 09:32 - Aniket Upganlawar

+1 for 0.9

#### #24 - 2009-06-24 09:49 - Paul Rivier

this feature needs more work than what I provided in my previous patch. I would like to find some time in the coming months to polish it.
Jean-Philippe, before I get started with rewriting it for current trunk, do you have some opinion or advises on that feature ?

#### #25 - 2009-07-06 15:00 - Vitaliy Ischenko

*- File 33-add_versions_inheritance-0.4.0.patch added*

I've tryed to update patch for trunk
it has the same problems as the previous patch: just filter sensible information from every place where it can appear

also it ads some usability improvements to selecting inherited versions in every place i managed to find :)

#### #26 - 2009-09-02 02:23 - Eric Davis

I'm working on a patch that might work for this. It uses all versions in a specific branch of a project's hierarchy. That means you can assign an issue to a grandparent's version or any children. Quick example:

- Project A
  - Project AA
- Project B
  - Project BA
  - Project BB
    - Project BBB
  - Project BC
- Project C

- Issue on Project A - can be assigned to a version on Project A or AA
- Issue on Project B - can be assigned to a version on Project B, BA, BB, or BBB
- Issue on Project BB - can be assigned to a version on Project B, BA, BB, or BBB
- Issue on Project C - can be assigned to a version on Project C only

There's still quite a bit to do (permissions, visibility, turn on/off) but it's working good so far.

#### #27 - 2009-09-02 10:51 - Paul Rivier

Hi Eric,

> There's still quite a bit to do (permissions, visibility, turn on/off) but it's working good so far.

thank you for working on this, how to you plan to handle the situation of a project being moved elsewhere in the hierarchy, with issues fixed to non-visible versions is the new place. On my part I just added a validation to ensure this can't happen, but something like restricting the dropdown list of target parent project could maybe help.

#### #28 - 2009-09-02 23:40 - Eric Davis

Here's a question: I'm using the term inherited_versions to describe the versions on children and parent projects. Does that make sense or is there a better term I can use?

#### #29 - 2009-09-03 00:50 - Eric Davis

*- Category set to Roadmap*

*- Status changed from New to 7*

*- Assignee changed from Jean-Philippe Lang to Eric Davis*

Commenting on a few design questions since I'm working on this right now.

Paul Rivier wrote:

> > Maybe we could disable the link to the version (just display its name) if the user is not allowed to view it.
>
> Because the name itself is a sensitive information (like "We buy company X"), my point of view is that even the name should be hidden. What do you think ?

I think it should mask the link and the name. So an unauthorized user would see something like "Version: <not authorized>". If the unauthorized user has permission to edit the issue, the select field should include that version and it's id (since it's selected) but still keep the name masked. This would also need to be changed in the Journals, since they will show the Version name "Target version set to <not authorized>". I'd be open to another UI suggestion if we can make sure private data is hidden.

Paul Rivier wrote:

> thank you for working on this, how to you plan to handle the situation of a project being moved elsewhere in the hierarchy, with issues fixed to non-visible versions is the new place. On my part I just added a validation to ensure this can't happen, but something like restricting the dropdown list of target parent project could maybe help.

Good point. Right now my code keeps the link to the original version (still working on the core feature). Moving an **Issue** out of the project hierarchy removes the Target Version. Maybe if a **Project** is moved out of the project hierarchy, it should remove the Target Versions that aren't accessible in the new hierarchy (i.e. only keep assignments to Versions on itself or projects it can reach). Very worst case is; an issue's version is removed, the project members are notified of the event, and the change is logged as a Journal note to the issue. What do you think?

#### #30 - 2009-09-03 06:55 - Bill Tihen

Hi Eric,

It makes sense to me that the Versions should be removed if the task or project moves out of the scope of a version, but it would be nice if to keep the version if it is still within the versions scope -- for example.

```
Project A - has v A1
  Project AA
  Project AB
    Project mobile - assigned version A1

moves to:

Project A - vA1
  Project AA
    Project mobile -- assigned version A1
  Project AB

but if it moves to:

Project Mobile -- remove version
Project A - has v A1
  Project AA
  Project AB
```

If it can be aware of version scope and remove it when it leaves the scope that would be the best in my opinion.

> Maybe if a Project is moved out of the project hierarchy, it should remove the Target Versions that aren't accessible in the new hierarchy (i.e. only keep assignments to Versions on itself or projects it can reach). Very worst case is; an issue's version is removed, the project members are notified of the event, and the change is logged as a Journal note to the issue. What do you think?

#### #31 - 2009-09-03 06:57 - Bill Tihen

PS -- this seems like a reasonable privacy solution.

> I think it should mask the link and the name. So an unauthorized user would see something like "Version: <not authorized>". If the unauthorized user has permission to edit the issue, the select field should include that version and it's id (since it's selected) but still keep the name masked. This would also need to be changed in the Journals, since they will show the Version name "Target version set to <not authorized>". I'd be open to another UI suggestion if we can make sure private data is hidden.

#### #32 - 2009-09-03 09:24 - Paul Rivier

Hi Eric,

> Here's a question: I'm using the term inherited_versions to describe the versions on children and parent projects.  Does that make sense or is there a better term I can use?

I'd rather use the term **related_versions** since "inheritance" means "access to parent data". Also, the term "accumulation" means "access to children data". Maybe those two mecanisms should be kept as separate options at the project level, since they means two different things. What do you think Eric ?

> So an unauthorized user would see something like "Version: <not authorized>".

I'd prefer a simple dash "-" instead of "<not authorized>" but that a faily minor detail.

Also, after a year of use of my inheritance patch, I'm a bit puzzled about this feature. It is not something easy to design, maintain and understand given the current state of things in redmine, i.e. with a project hierarchy that mainly acts as a placeholder to put a project and find it later. I feel it would make more sens to get this patch if we had stricter rules about permissions, roles, data propagation inside project hierarchy (both ways).

So I have been thinking about an other solution : a flag on version to make it either private (local to the project), shared (local to the hierarchy) or public (local to the whole instance). Then a simple interface in the project settings to "import" a foreign version, shared or public, so that it get accessible from the current project. What do you think about that ?

#### #33 - 2009-09-03 21:18 - Eric Davis

Bill Tihen wrote:

> It makes sense to me that the Versions should be removed if the task or project moves out of the scope of a version, but it would be nice if to

> keep the version if it is still within the versions scope -- for example.

I agree, that sounds like a reasonable behavior.

Paul Rivier wrote:

> I'd rather use the term **related_versions** since "inheritance" means "access to parent data". Also, the term "accumulation" means "access to children data". Maybe those two mecanisms should be kept as separate options at the project level, since they means two different things. What do you think Eric ?

**Related** could work (it's better than **Inherited**) but it sounds like comparing two versions. Like "Version 1.0 is **related** to Version 1.1". What about **Available**? "Version 1.0 is **available** for Project A".

> > So an unauthorized user would see something like "Version: <not authorized>".
>
> I'd prefer a simple dash "-" instead of "<not authorized>" but that a faily minor detail.

I'd prefer to be explicit so the user knows why it looks different. It will be easy to change later, it's in the translation file.

> Also, after a year of use of my inheritance patch, I'm a bit puzzled about this feature. It is not something easy to design, maintain and understand given the current state of things in redmine, i.e. with a project hierarchy that mainly acts as a placeholder to put a project and find it later. I feel it would make more sens to get this patch if we had stricter rules about permissions, roles, data propagation inside project hierarchy (both ways).

Could you explain this further, I'm not seeing any limitations so far.

> So I have been thinking about an other solution : a flag on version to make it either private (local to the project), shared (local to the hierarchy) or public (local to the whole instance). Then a simple interface in the project settings to "import" a foreign version, shared or public, so that it get accessible from the current project. What do you think about that ?

I don't like the "import" idea but the rest of this sounds like a mix of the options from Jean-Philippe above. I'm going to be working on the "turn on/turn off" configuration soon so I'll see if that suggestion would work easily.

Jean-Philippe, anything you'd like to add?

#### #34 - 2009-09-03 23:33 - Eric Davis

How about the term "Shared Version"? That will map easily to the "No sharing", "Share with hierarchy", or "Share with any project" options.

#### #35 - 2009-09-03 23:46 - Eric Davis

*- File 465-shared-versions-work-in-progress.patch added*

I've just pushed the current code to a branch on Github and also attached a patch here. It's still a work in progress but I'd appreciate any feedback anyone has. It's missing:

- Options to turn on/off the sharing/inheritance
- Versions will stick if an Issue or Project is moved out of scope

Branched from r2845

#### #36 - 2009-09-07 21:21 - Eric Davis

*- % Done changed from 0 to 100*

Just pushed up the final bits to Github. This version includes four changes:

1. When an issue with a Version is moved, it will unassign the Version if it's out of scope.
2. When a project is moved, it will unassign the Versions for all issues that are out of scope. This will include issues that are using the moved project's version.
3. Added a field to each Version that will let you setup how it should be shared. Default is "none" but there are "Parent and child project" and "Systemwide" sharing options. (**requires a database migration**)
4. Renamed inherited_versions to shared_versions

With such a large change, I'd like to get some feedback and a few reviews of this before I commit it. I'm looking for any bugs, edge cases, or weird behavior.

#### #37 - 2009-09-08 22:18 - ciaran jessup

Eric Davis wrote:

> Just pushed up the final bits to Github. This version includes four changes:
>
> 1. When an issue with a Version is moved, it will unassign the Version if it's out of scope.
> 2. When a project is moved, it will unassign the Versions for all issues that are out of scope. This will include issues that are using the moved project's version.
> 3. Added a field to each Version that will let you setup how it should be shared. Default is "none" but there are "Parent and child project" and "Systemwide" sharing options. (**requires a database migration**)
> 4. Renamed inherited_versions to shared_versions
>
> With such a large change, I'd like to get some feedback and a few reviews of this before I commit it. I'm looking for any bugs, edge cases, or weird behavior.

Looking good so far, any areas in particular you foresee being edgy. Done a few searches, added and removed some versions. Looks pretty fricken awesome tbh! :)

#### #38 - 2009-09-10 02:49 - Eric Davis

Biggest areas that might hide some bugs are:

- Permisisons
  - User A can't see Project P
  - But can see Issue #abc which has a Fixed Version which is on Project P
- Moving issues and projects to different parts of the project tree (e.g. to the root level, to a subproject) and checking how the versions follow

#### #39 - 2009-09-10 09:21 - yusuke kokubo

+1

This is very important feature to me.
I hope the patch will be marge to trunk as soon as possible.

#### #40 - 2009-09-15 02:25 - Alexander Pánek

+1

Wondering why this isn't already included for a loooong time. :)

#### #41 - 2009-09-15 09:55 - jason axelson

+1

So is this code going to be merged into the trunk?

#### #42 - 2009-09-15 23:20 - Reavis Sutphin-Gray

+1

I can't wait to see this in trunk!

#### #43 - 2009-09-17 15:41 - Paul Rivier

Hi Eric, thanks for working on that.

> 1. When an issue with a Version is moved, it will unassign the Version if it's out of scope.
> 2. When a project is moved, it will unassign the Versions for all issues that are out of scope. This will include issues that are using the moved project's version.
> 3. Added a field to each Version that will let you setup how it should be shared. Default is "none" but there are "Parent and child project" and "Systemwide" sharing options. (**requires a database migration**)

What happens if an issue is planned for a version while its "systemwide", then this version becomes "local only" ?

#### #44 - 2009-10-07 18:06 - Eric Davis

Paul Rivier wrote:

> What happens if an issue is planned for a version while its "systemwide", then this version becomes "local only" ?

Good question. I would expect when a version's sharing is changed from systemwide to none (i.e. local), then each non-local issue is removed from that version. I already have a method that will check each issue and update it's version if it's invalid. I'm using for when a project is moved, so I can easily add that for Versions also.

```
# class Issue

  # Update all issues so their versions are not pointing to a
  # fixed_version that is outside of the issue's project hierarchy.
  #
  # OPTIMIZE: does a full table scan of Issues with a fixed_version.
  def self.update_fixed_versions_from_project_hierarchy_change
    Issue.all(:conditions => ['fixed_version_id IS NOT NULL'],
              :include => [:project, :fixed_version]
              ).each do |issue|
      next if issue.project.nil? || issue.fixed_version.nil?
      unless issue.project.shared_versions.include?(issue.fixed_version)
        issue.init_journal(User.current)
        issue.fixed_version = nil
        issue.save
      end
    end
  end
```

#### #45 - 2009-10-08 09:09 - Franco Campanale

Thanks for your fantastic work!
I'm using Redmine 0.8.5, when I'll be able to use this feature? (I would like to simply assign an issue of a subproject to a father project version)
What can I do to use it now?

Cheers

#### #46 - 2009-10-29 13:53 - Marco Stolze

At first .. sorry for my bad english .. and thanks for the impressive work!

Before i saw this issue my thoughts about the needs for use in our company were in the same direction but in a simpler way.

My understanding of "Related versions":
Redmine supports relations between a parent and his child projects so why it don't supports relations between a version of the parent project and versions of child projects?
When i create or modify a version definition in a child project it should be possible to define a relation to a version of the parent project.

```
Project: Main product suite
  |
  -- Version: Product 4.3.2
  |
  ChildProject: Module 1
    |
    -- Version: Module 3.2.1.234
-
Version: Product 4.3.2
  |
  -- Version: Module 3.2.1.234
```

In the roadmap of the parent project now i also should see for "Version: Product 4.3.2" the issues of "Version: Module 3.2.1.234". ... and .. if the child project has childs also the issues of the childs child project related versions to "Version: Module 3.2.1.234" ...
So I can represent multiple views on the relationship between version and issue:

- The developer has a documentation for the changes and what to do for the next version of "his" project.
- Possibly a documentation about the relationship of a product version an a modules file version.
- The manager has an overview over the complete product suite or the "sprint backlogs" over multiple projects of a team etc.
- ...

When the relationship between a parent and a child project will be released also the relationships between the parent's versions and the child's versions should be released.

In my opinion security issues and changes to the domain model are moderate.
Usability ?
Jean-Philippe, is this a usable model ?

#### #47 - 2009-11-14 17:05 - Patrick Hurrelmann

Eric Hulser, what is your estimation on when this will hit the tree?

This is one of the last missing major features for 0-9 imho, now
that ticket permissions and closed versions are already in.

#### #48 - 2009-11-14 19:11 - Eric Davis

Going to try to update and merge it in today or tomorrow, I just got some higher priority bugs I have to fix first.

#### #49 - 2009-11-25 20:59 - Eric Davis

Logging a bug report that was emailed to me by Corry Haines:

> In the '20090907170038_populate_version_shared.rb' file, located in 'db/migrate' you use double quotes to specify the value to set the shared column to. This is technically incorrect in SQL, as this indicates that you want to set the shared column to be equal to the none column. While this may work in other DBs, postgres does not allow it.

```
diff --git a/db/migrate/20090907170038_populate_version_shared.rb b/db/migrate/20090907170038_populate_version
_shared.rb
index b6aafc2..114eb14 100644
--- a/db/migrate/20090907170038_populate_version_shared.rb
+++ b/db/migrate/20090907170038_populate_version_shared.rb
@@ -1,6 +1,6 @@
 class PopulateVersionShared < ActiveRecord::Migration
   def self.up
-    Version.update_all('shared = "none"', ['shared IS NULL OR shared = ?',''])
+    Version.update_all('shared = \'none\'', ['shared IS NULL OR shared = ?',''])
   end

   def self.down
```

#### #50 - 2009-11-30 15:26 - Paul Merlin

This issue is not attached any version but says it's 100% done.

Will this be in the 0.9 release ?

Regards

/Paul

#### #51 - 2009-12-03 22:43 - Eric Davis

*- File 0001-Added-Version-sharing.-465.patch added*

*- % Done changed from 100 to 0*

I've updated the "patch" for Paul Rivier's suggestion, Corry Haines's bug report, and the recent changes for closing versions (#1245). I think this is ready for a code review and then I can commit it.

#### #52 - 2009-12-04 16:28 - Jean-Philippe Lang

Here are a few points I'd like to discuss before this patch gets checked in:

- Hierarchy sharing level:
  - I was expecting versions to be shared with subprojects (as initially requested here, or in #2063), not with parent projects. What is the use case behind this ?
  - If there is a **real** need, we should at least add a 'Child projects' sharing level
- System sharing level:
  - It's incompatible with an environment where projects are independent. That would allow a manager to "pollute" other projects by setting versions of his projects as systemwide shared
  - I think it's unnatural now that we have unlimited project nesting
- ApplicationHelper#format_version_name: if a version is **shared**, why should we prepend its name with a project name ?
- Version.update_fixed_versions_from_project_hierarchy_change can be **really** slow and is called every time a version is saved. When changes actually happen, be aware that a bunch of emails can also be sent.
- If 'Child and parents' sharing level is preserved, Version.update_fixed_versions_from_project_hierarchy_change should be called when a project is archived since its versions can be shared with its parents that are still active
  Note that when archiving a project tree, issues of these projects should not be unassigned (archive/unarchive should be reversible).
- What about adding Version#visible? (like Issue#visible?) rather than using user.allowed_to?(...) everywhere we want to check if the version is visible?
- Project#shared_versions should better be a :has_many association, so we can do nice things like project.shared_versions.find(1) instead of project.shared_versions.find {|v| v.id.to_s == 1 } that runs 3 SQL queries and loads all the shared versions.
- Last but not least, is there a **real** need to hide the name of a version that belongs to a project the user doesn't have access to? Hiding something that is shared seems a bit odd to me. I already asked this question above but I don't think that the example ("We buy company X") is realistic.

#### #53 - 2009-12-04 17:39 - Felix Schäfer

My 0,02€ on this one (and I haven't tried the patch, those are only comments base on previous discussion):

1. I'd really welcome having the versions from parent projects in child projects, or at least the option to allow those, but I can't think of a use case where having versions from a child project in the parent project would be necessary, and even then, why not just create the version in the parent project instead of in the child project?
2. Not sure if system-wide versions are really needed, but even so, it should just be a global enumeration like document categories or whatever.
3. I wouldn't automatically prepend the project name, if you need it, you still can call the version "$PROJECT - $VERSION".
4. Not sure about the name-hiding, I don't think it's really necessary, but maybe you could just make it an option or a permission?

#### #54 - 2009-12-04 17:52 - Corry Haines

Felix Schäfer wrote:

> 1. I'd really welcome having the versions from parent projects in child projects, or at least the option to allow those, but I can't think of a use case where having versions from a child project in the parent project would be necessary, and even then, why not just create the version in the parent project instead of in the child project?

Inheriting projects in the parent is useful when the parent is meant to be a "shell" project. For example, if the root of the tree is called "Engineering" and there are multiple sub-projects, then it is useful for management to see all of the versions in the root project.

As for creating versions in the parent rather than the child, this comes down to a permissions issue. It is possible that only a few people would have access to the shell, while adding new version may be more ad-hoc.

#### #55 - 2009-12-04 19:03 - Felix Schäfer

Ok, that would make sense.

Then I'll also revise my opinion on the project-name-prefixed version names: I think a sensible default would be to not prepend the project name if the version comes from a parent project, and to prepend it if it comes from a child project.

#### #56 - 2009-12-04 19:11 - Jean-Philippe Lang

Corry Haines wrote:

> Inheriting projects in the parent is useful when the parent is meant to be a "shell" project. For example, if the root of the tree is called "Engineering" and there are multiple sub-projects, then it is useful for management to see all of the versions in the root project.

I agree that we should be able to have an overview all the subprojet versions from the root project, for example on the roadmap. But it doesn't meen that we want to create issues in the parent project that are assigned to a subproject version.

#### #57 - 2009-12-04 21:26 - Jean-Philippe Lang

*- Target version set to 0.9.0*

#### #58 - 2009-12-05 01:30 - Eric Davis

Jean-Philippe Lang wrote:

> Here are a few points I'd like to discuss before this patch gets checked in:
>
> - Hierarchy sharing level:
>
> - I was expecting versions to be shared with subprojects (as initially requested here, or in #2063), not with parent projects. What is the use case behind this ?

Corry Haines summarized that.  I don't see any reason to limit a child version to only go deeper (e.g. child > sub-child) and not shallower (e.g. child to it's parent).

- If there is a **real** need, we should at least add a 'Child projects' sharing level

I could see the usefulness of a Child sharing.  Probably should include the current project and it's children though, not just children.

- System sharing level:

- It's incompatible with an environment where projects are independent. That would allow a manager to "pollute" other projects by setting versions of his projects as systemwide shared
- I think it's unnatural now that we have unlimited project nesting

I'd disagree, it was developed for an environment where many projects are independent but with some common areas.  Another use case is by

separating Redmine's projects based on function.  For example an IT organization would have projects like:

- Servers
  - Firewall servers
  - Database servers
  - File servers
- Networking
  - Switches
    - Cisco switches
- Datacenter 1
  - Version - Expansion 2009
  - Version - Expansion 2010

The Servers and Networking projects would have issues and documentation about the different hardware.  But Expansion 2009 needs a new firewall (Firewall project) and a database (Database project).  By having the version "Expansion 2009" be systemwide, the issues for setting up a firewall can be in the Firewall project along with the firewall team and it's documentation.

- ApplicationHelper#format_version_name: if a version is **shared**, why should we prepend its name with a project name ?

A Version's name is unique based on the project.  So you could potentially have multiple versions in a select box and not know which is which:

- Version 0.1.0 (From Project A)
- Version 0.1.0 (From Project B)

- Version.update_fixed_versions_from_project_hierarchy_change can be **really** slow and is called every time a version is saved. When changes actually happen, be aware that a bunch of emails can also be sent.

100% agree.  Ideally we would background that action but we don't have a backgrounding system yet.  I think the email notifications should still happen, in order to notify the members of each project that something changed (important if the version change was on a different project).

- If 'Child and parents' sharing level is preserved, Version.update_fixed_versions_from_project_hierarchy_change should be called when a project is archived since its versions can be shared with its parents that are still active
  Note that when archiving a project tree, issues of these projects should not be unassigned (archive/unarchive should be reversible).

Good point, I missed that action.

- What about adding Version#visible? (like Issue#visible?) rather than using user.allowed_to?(...) everywhere we want to check if the version is visible?

I agree, that would be more clear.

- Project#shared_versions should better be a :has_many association, so we can do nice things like project.shared_versions.find(1) instead of project.shared_versions.find {|v| v.id.to_s == 1 } that runs 3 SQL queries and loads all the shared versions.

That's what I started with originally but it wouldn't work because systemwide and hierarchy versions are not associated with the project at all.  We could improve the performance if we write some custom sql for that method or as the association but I went with clear code over the fastest code for this patch.

- Last but not least, is there a **real** need to hide the name of a version that belongs to a project the user doesn't have access to? Hiding something that is shared seems a bit odd to me. I already asked this question above but I don't think that the example ("We buy company X") is realistic.

I don't know.  In my use cases, I don't think it's needed but it seemed like Paul River might have a more realistic example.

Thoughts?

#### #59 - 2009-12-05 01:36 - Eric Davis

I forgot to mention: I have a mostly functional rewrite of the Gantt chart using this that makes Redmine's Gantt page draw outline looking charts.  Should I create a patch to start the review process for the Gantt rewrite and just assume this is committed for now?  If this feature's API changes, I can update the Gantt patch.

#### #60 - 2009-12-05 10:15 - Jean-Philippe Lang

I'm refactoring the patch but found that if you set a version shared with 'Parent and child projects', it's also shared with sibling projects (and in fact with all the descendants of the root project).
Is it just a misleading label (maybe 'Project tree' would be better in this case) or an undesired behaviour?

#### #61 - 2009-12-05 10:21 - Jean-Philippe Lang

Eric Davis wrote:

> The Servers and Networking projects would have issues and documentation about the different hardware. But Expansion 2009 needs a new firewall (Firewall project) and a database (Database project). By having the version "Expansion 2009" be systemwide, the issues for setting up a firewall can be in the Firewall project along with the firewall team and it's documentation.

Wouldn't it make sense if servers and networking were subprojects of the datacenter?

#### #62 - 2009-12-06 11:29 - Jean-Philippe Lang

*- Assignee changed from Eric Davis to Jean-Philippe Lang*

*- Resolution set to Fixed*

Patch committed with heavy changes in r3123.
2 additional sharing levels were added. See the commit log for more info.

#### #63 - 2009-12-07 04:20 - Mischa The Evil

Jean-Philippe Lang wrote:

> Patch committed with heavy changes in r3123.

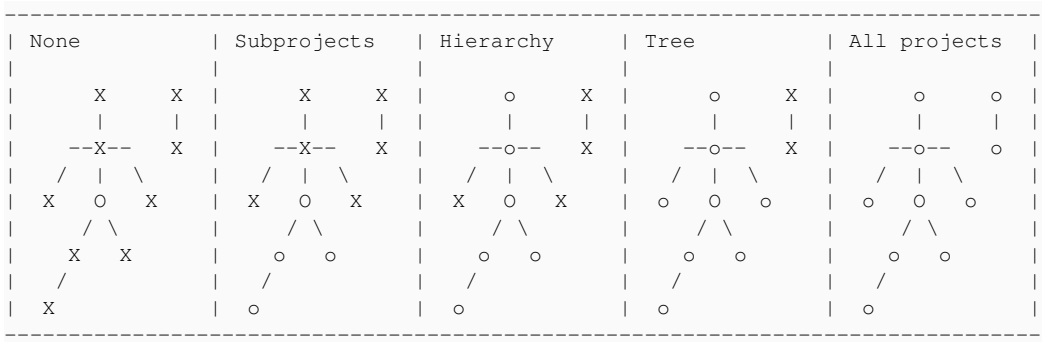
I've wrapped-up a file to test the inheritance-scope. I've pasted the results in a public gist. It's located at http://gist.github.com/250572.
Quickly reviewing the results gives me a feeling it's not as clear as the changeset comment of r3123 says. I'll review it in more depth this week.

#### #64 - 2009-12-07 20:23 - Jean-Philippe Lang

The 'subprojects' sharing level is fixed in r3130.

#### #65 - 2009-12-07 20:48 - Jean-Philippe Lang

Here are some examples that illustrate sharing levels:




```
O = project with the shared version
o = projects that can assign issues to the shared version
X = projects that can't assigne issues to the shared version
```

#### #66 - 2009-12-07 21:14 - Eric Davis

Jean-Philippe Lang wrote:

> I'm refactoring the patch but found that if you set a version shared with 'Parent and child projects', it's also shared with sibling projects (and in fact with all the descendants of the root project).
> Is it just a misleading label (maybe 'Project tree' would be better in this case) or an undesired behaviour?

Correct, it was mislabeled.  It would be the "Tree" option you committed.

Jean-Philippe Lang wrote:

> Patch committed with heavy changes in r3123.
> 2 additional sharing levels were added. See the commit log for more info.

Great.  I'll do a code review of it soon. I see you converted Project#shared_versions to use some custom sql, that should help the performance of my patch.

I'll try to do a blog post about this with some screenshots of the UIs to the redmineblog.com.  This feature could be confusing without some

documentation (though your ASCII Art looks good too).

#### #67 - 2009-12-07 22:11 - Mischa The Evil

Jean-Philippe Lang wrote:

> The 'subprojects' sharing level is fixed in r3130.

Great! That was indeed the sharing level which was yet unclear to me. Now I see why... :) Thanks for the speedy fix!

#### #68 - 2009-12-07 22:20 - Mischa The Evil

Eric Davis wrote:

> I'll try to do a blog post about this with some screenshots of the UIs to the redmineblog.com. This feature could be confusing without some documentation (though your ASCII Art looks good too).

This is yet another feature in 0.9.0 which indeed needs proper documentation to get understood well. I also like Jean-Philippe's ASCII art very much and I think it is a good way to explain the different levels of sharing.

I'll also try to update the Redmine Guide soon with the info provided here and in the changeset comment to make sure this more-complex, but powerfull feature is documented well and proper.

#### #69 - 2009-12-07 23:05 - Mischa The Evil

As discussed over IRC with Eric Davis I'd like to propose an enhancement to this feature:

Currently it's not possible to get a per-project overview of the available versions. It would be nice if all the projects available versions can be displayed in the project settings (at */projects/xxx/settings/versions*). Though it seems the best to me to keep inherited versions not-editable to prevent confusion. Maybe the inherited versions can be displayed in a differenciated way to distinct them from the own project versions?

#### #70 - 2009-12-07 23:14 - Eric Davis

Mischa The Evil:

I pulled your last request out and moved it to #4357.  I agree it would be useful but I don't want it to prevent this one from finishing up.

#### #71 - 2009-12-08 11:58 - Rickard Åberg

Did anyone here have an issue with the "Target version" combobox after this update. (I guess it's related to this change). The drop-down is filled with the html <option value>-tags and the selections don't have any affect. It's like the option values have become labels, and the values are probably NULL or illegal.

#### #72 - 2009-12-08 13:00 - Jean-Philippe Lang

*- Category changed from Roadmap to Projects*

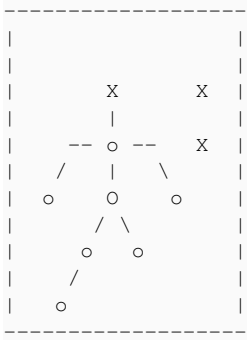
*- Status changed from 7 to Closed*

I'm closing this feature request.

Richard, I have no issue with the "Target version" bombo. Please fill a detailed defect report if needed.

#### #73 - 2010-02-23 17:59 - Nanda P

- It will be nice to have another level of Version sharing.



Versions shared between Immediate parent & child

#### #74 - 2010-10-19 17:56 - Robin McKenzie

We have 0.9.3, but I this feature doesn't seem to be included, and I can't find any info in the Redmine wiki on how to enable it - can anyone help please?

#### #75 - 2010-10-19 18:10 - Robin McKenzie

Robin McKenzie wrote:

> We have 0.9.3, but I this feature doesn't seem to be included, and I can't find any info in the Redmine wiki on how to enable it - can anyone help please?

My mistake, it's there, at the level of each individual Version.

Could I make a request to have an option at the level of the parent project which allows all Versions to be inherited, rather than on a per-Version basis?  Thanks.

#### #76 - 2010-10-19 18:28 - Colan Schwartz

Robin McKenzie wrote:

> Could I make a request to have an option at the level of the parent project which allows all Versions to be inherited, rather than on a per-Version basis?  Thanks.

This issue is closed.  Please open new issues for new feature requests.

#### #77 - 2010-11-16 08:04 - James Selvakumar

Robin McKenzie wrote:

> Robin McKenzie wrote:
>
> > We have 0.9.3, but I this feature doesn't seem to be included, and I can't find any info in the Redmine wiki on how to enable it - can anyone help please?
>
> My mistake, it's there, at the level of each individual Version.

I too like Robin couldn't figure that out on my first sight. But it's a great feature and thanks guys for this wonderful product!

### Files

| | | | |
|---|---|---|---|
| veersion_inheritance.patch | 22.5 KB | 2008-06-27 | Paul Rivier |
| 33-add_versions_inheritance-0.4.0.patch | 27.8 KB | 2009-07-06 | Vitaliy Ischenko |
| 465-shared-versions-work-in-progress.patch | 30.9 KB | 2009-09-03 | Eric Davis |
| 0001-Added-Version-sharing.-465.patch | 80.6 KB | 2009-12-03 | Eric Davis |