

Redmine - Defect #34108

"Is duplicate of" and "Has duplicate" issue relations can be a circular reference

2020-10-13 05:50 - Go MAEDA

Status:	New	Start date:	
Priority:	Normal	Due date:	
Assignee:		% Done:	0%
Category:	Issues	Estimated time:	0.00 hour
Target version:	Candidate for next minor release	Affected version:	
Resolution:			

Description

You can make two issues circular reference with the following steps:

1. Suppose that there are two issues A and B
2. Open the issue A and add a "Is duplicate of" relation to issue B
3. Open the issue B and add a "Is duplicate of" relation to issue A

After the above operation, each of those two issues will have two relations against the opponent, "Has duplicate and Is duplicate of" (see the screenshot below). Such logically wrong relations should not be created.

[circular-relation.png](#)

The circular reference causes a serious problem when you try to close one of those issues. If you attempt to close the issue, Redmine raises SystemStackError exception and hangs.

```
SystemStackError (stack level too deep):
```

```
lib/plugins/acts_as_customizable/lib/acts_as_customizable.rb:92:in `detect'
lib/plugins/acts_as_customizable/lib/acts_as_customizable.rb:92:in `block in custom_field_values'
lib/plugins/acts_as_customizable/lib/acts_as_customizable.rb:81:in `collect'
lib/plugins/acts_as_customizable/lib/acts_as_customizable.rb:81:in `custom_field_values'
app/models/journal.rb:193:in `start'
app/models/journal.rb:77:in `initialize'
app/models/issue.rb:823:in `init_journal'
app/models/issue.rb:1859:in `block in close_duplicates'
app/models/issue.rb:1851:in `each'
app/models/issue.rb:1851:in `close_duplicates'
app/models/issue.rb:215:in `create_or_update'
app/models/issue.rb:1862:in `block in close_duplicates'
app/models/issue.rb:1851:in `each'
app/models/issue.rb:1851:in `close_duplicates'
app/models/issue.rb:215:in `create_or_update'
app/models/issue.rb:1862:in `block in close_duplicates'
app/models/issue.rb:1851:in `each'
app/models/issue.rb:1851:in `close_duplicates'
app/models/issue.rb:215:in `create_or_update'
app/models/issue.rb:1862:in `block in close_duplicates'
app/models/issue.rb:1851:in `each'
app/models/issue.rb:1851:in `close_duplicates'
app/models/issue.rb:215:in `create_or_update'
app/models/issue.rb:1862:in `block in close_duplicates'
app/models/issue.rb:1851:in `each'
app/models/issue.rb:1851:in `close_duplicates'
app/models/issue.rb:215:in `create_or_update'
app/models/issue.rb:1862:in `block in close_duplicates'
app/models/issue.rb:1851:in `each'
app/models/issue.rb:1851:in `close_duplicates'
app/models/issue.rb:215:in `create_or_update'
app/models/issue.rb:1862:in `block in close_duplicates'
app/models/issue.rb:1851:in `each'
```

History

#1 - 2020-10-13 09:14 - Yuichi HARADA

- File *fixed-34108.patch* added

I made the following patch with reference to the modification of [#27663](#).

```
diff --git a/app/models/issue_relation.rb b/app/models/issue_relation.rb
index d0a66ba40..e0075cb0a 100644
--- a/app/models/issue_relation.rb
+++ b/app/models/issue_relation.rb
@@ -239,6 +239,10 @@ class IssueRelation < ActiveRecord::Base
   issue_from.blocks? issue_to
   when 'blocks'
     issue_to.blocks? issue_from
+   when 'duplicated'
+     self.class.where(issue_from_id: issue_from, issue_to_id: issue_to, relation_type: TYPE_DUPLICATES).exists?
+   when 'duplicates'
+     self.class.where(issue_from_id: issue_to, issue_to_id: issue_from, relation_type: TYPE_DUPLICATES).exists?
   when 'relates'
     self.class.where(issue_from_id: issue_to, issue_to_id: issue_from).present?
   else
```

#2 - 2020-10-14 07:49 - Yuichi HARADA

- File *fixed-34108-v2.patch* added

The registration data may contain Issues that you have already circular dependency. To solve this, add the following patch to [fixed-34108.patch](#).

```
diff --git a/app/models/issue.rb b/app/models/issue.rb
index 8c3146137..21aed394b 100644
--- a/app/models/issue.rb
+++ b/app/models/issue.rb
@@ -1848,6 +1848,14 @@ class Issue < ActiveRecord::Base
   # Closes duplicates if the issue is being closed
   def close_duplicates
     if Setting.close_duplicate_issues? && closing?
+     # Check that there are no circular dependency of relationships
+     relations_to.where(:relation_type => IssueRelation::TYPE_DUPLICATES).each do |relation|
+       unless relation.valid?
+         errors.add :base, relation.errors.full_messages.first
+         throw :abort
+       end
+     end
+     end
+
     duplicates.each do |duplicate|
       # Reload is needed in case the duplicate was updated by a previous duplicate
       duplicate.reload
```

#3 - 2020-10-15 16:07 - Go MAEDA

- Target version set to Candidate for next minor release

#4 - 2020-12-20 10:32 - Go MAEDA

Yuichi HARADA wrote:

I made the following patch with reference to the modification of [#27663](#).

[...]

Thank you for writing the patch. But I found that [fixed-34108-v2.patch](#) does not cover the situation that 3 or more issues make circular dependency.

- Issue 101 ---(has duplicate)---> Issue 102
- Issue 102 ---(has duplicate)---> Issue 103
- Issue 103 ---(has duplicate)---> Issue 101

#5 - 2020-12-23 03:57 - Yuichi HARADA

- File *fixed-34108-v3.patch* added

Go MAEDA wrote:

Thank you for writing the patch. But I found that [fixed-34108-v2.patch](#) does not cover the situation that 3 or more issues make circular dependency.

- Issue 101 ---(has duplicate)---> Issue 102
- Issue 102 ---(has duplicate)---> Issue 103
- Issue 103 ---(has duplicate)---> Issue 101

Thank you for pointing this out. I have fixed it to cover situations where three or more issues cause circular dependencies.

```
diff --git a/app/models/issue.rb b/app/models/issue.rb
index 8ba261bb1f..a8810611cb 100644
--- a/app/models/issue.rb
+++ b/app/models/issue.rb
@@ -1930,6 +1930,13 @@ class Issue < ActiveRecord::Base
  # Closes duplicates if the issue is being closed
  def close_duplicates
    if Setting.close_duplicate_issues? && closing?
+   # Check that there are no circular dependency of relationships
+   duplicate_issue_ids = circular_dependency_duplicate_issue_ids(duplicates)
+   if duplicate_issue_ids.include?(self.id)
+     self.errors.add :base, :circular_dependency
+     throw :abort
+   end
+
    duplicates.each do |duplicate|
      # Reload is needed in case the duplicate was updated by a previous duplicate
      duplicate.reload
@@ -1946,6 +1953,17 @@ class Issue < ActiveRecord::Base
  end
end

+ def circular_dependency_duplicate_issue_ids(issues, duplicate_ids = [])
+   issues.each do |issue|
+     next if duplicate_ids.include?(issue.id)
+
+     duplicate_ids << issue.id
+     duplicate_ids.concat(circular_dependency_duplicate_issue_ids(issue.duplicates, duplicate_ids))
+     duplicate_ids.uniq!
+   end
+   duplicate_ids
+ end
+
  # Make sure updated_on is updated when adding a note and set updated_on now
  # so we can set closed_on with the same value on closing
  def force_updated_on_change
```

Files

circular-relation.png	17 KB	2020-10-13	Go MAEDA
fixed-34108.patch	1.79 KB	2020-10-13	Yuichi HARADA
fixed-34108-v2.patch	3.93 KB	2020-10-14	Yuichi HARADA
fixed-34108-v3.patch	5.37 KB	2020-12-23	Yuichi HARADA