EDIT - feature request #4305

newly created entities must stay editable even if a user only has the permission to create them

08/06/2014 08:38 AM - Andreas Kohlbecker

Status:	In Progress	**Start date:**	
Priority:	Priority10	**Due date:**	
Assignee:	Andreas Kohlbecker	**% Done:**	40%
Category:	cdmlib	**Estimated time:**	0:00 hour
Target version:	Reviewed Next Major Release		
Severity:	normal		

Description

Users which only have the authority to create an entity CREATE are blocked from changing it afterwards since they miss the UPDATE authority. In this situation it is no possible to change typos or to do other minor fine tuning after the new entity has been saved for the first time.

To circumvent this situation of being locked out in the middle of the actual creation process we discussed the below described solutions. A full solution needs to solve the following two detail problems, which should be considered separately:

1. **Extended create problem**
2. **Multiple referencing objects problem.**

Concepts

A) continuedCreationTimespan

Once a 'Editor' has created a new entity it is considered being still in the process of being created for some time even if the entity has already been saved to the database. This means that the Authority CREATE implies UPDATE as long as

@now - entity.created < continuedCreationTimespan@}

AK: This approach is not appropriate in the context of applications like phycobank where entity must stay editable as long as the registration process is not completed, this can take several weeks.

B1) no further referencing object of other users

The entity stays editable as long as there are only referencing objects for which the **user equals to createdBy** and **updatedBy** in case this is set.

This solution requires to check the whole tree of referencing objects until the process reaches at least one object for which is not editable by the user. This solution might be too extensive in terms of computation, therefor it might be necessary to stop examining the tree after a certain timespan has been exceeded. On the other hand, cases in which this process is potentially running for a quite long time are most probably rather rare, because:

The process stops as soon a non editable referencing entity is reached. And in cases where the tree is big, there will be a lot of referencing object and the probability to find a non editable entity is rather high.

As stated in B2) it would be important to initially deny editing or saving the entity if there are more than one referencing object and to present the user a dialog with a list of all affected referencing objects, so that the user gets a profound and comprehensive knowledge on the consequences of the action.

B2) no further referencing objects

The entity stays editable as long as there is only one referencing object for which the **user equals to createdBy** and **updatedBy** in case this is set.
For how to deal with situation in which editing is denied due to multiple referencing object, please see below.

This is a pragmatic approach which covers most use-cases where a user creates e.g. a person which is referenced by exactly one reference.
Generally there are 2 reasons for not giving UPDATE rights to a user:

1. The user may change a record that is also used by data of *another* user and the danger is that (s)he changes it in an unwanted

way.
2. The user may change a record that is referenced multiple times and (s)he may not understand well enough that she changes the record for ALL references. May (s)he completely change the name of a taxon name author not realizing that this is correct for one given name but not for 5 other names that also use this name. This is the most dangerous case as data may become completely corrupted. Also checking if the user is the only user who references the record is not a solution here as the user may corrupt his/her own data as well as other's data. *In reality it seems to be much more important if an object is only referenced once or multiple times*. If referenced once the chance is small that data get corrupted while referenced multiple times the chance is high.

In current TaxEditor we already give exactly this warning (in red) at some places which helps a lot. We could replace the warning by a message that someone has no rights to edit such data.
This could also cover the A) case because in most cases relatively new records are not referenced multiple times.

dealing with 'edit denied' situations due to multiple referencing object

If the object is not editable in this sense the system may create a clone of the object. The clone no longer has no referencing objects and editing it would not have any side effects.

A the original of the cloned object could be marked with a Marker object denoting that a new version of this entity exists as clone. Users entering editors for objects referencing this original entity are informed about the new version, so that (sh)he can decide to use the new entity also in this case.

Using cloned objects, imposes a bit more complexity on the UI development. Let's take a look at the following example for illustrating this:

A user is editing Reference which has a Person as author, (s)he clicks on "Edit author". The Person is references by multiple objects, so the system creates a clone either during the save operation. After the Person editor is closed the Name still has the unmodified Person as author and must be notified about the fact that the edit operation has created a clone of the object. In turn of this notification the name editor switches the author relation do the new name. In case the user is clicking cancel on the name editor the change is not saved and the name still has the old author. Therefore a dialog should warn the user when leaving the editor if the data is unsaved.

C)

rejected due to X.3)

The entity stays editable as long as the **user equals to createdBy** and **updatedBy** in case it is set.

When performing the save operation, that is just when the user has clicked "save" and before the actual save transaction has been started, the user is presented a dialog when there are more than one referencing objects. This dialog lists all referencing objects and the user can select those for which the entity being saved should also be changed. For each object it will be checked if the user has the permission to update it (includes extended create). The user an select all objects for which (s)he has the permission. If the user selects all referencing objects the entity is just saved, otherwise a clone is being created which will be used for all selected objects.

This requires the following problems to be solved in case a clone is created:

- The property which holds the referenced object needs to be known or needs to be found.

- All objects need to be updated with the cloned object
  - what if the referenced object again has multiple referencing objects? Another referencing objects dialog needs to be opened and so one. How deep will this go into the object graph?

D) per instance UPDATE & DELETE permission.

After creating a new instance the user gets the permission to UPDATE & DELETE exactly this instance.
By this no further interpretation of the CREATE permission as extended create is necessary and the given permission can be revoked again.
The system will care for automatic assignment of the UPDATE & DELETE authority to the user whereas the new authorities will be direly associated
with the user. This has the advantage that the automatically given authorities can not be changed by a user manager via the editor, where only the authorities assigned to groups can be edited.

General considerations

X.1) limiting the amount of clones being created

refers to **B2)**, **C)**

A clone only needs to be created if the entity is changed significantly. For example changing a persons lifespan usually is not affecting the referencing objects and crating a clone is not needed. A *match* method could be used to perform this check:

```
boolean createClone = originalEntityVersion.matches(editedEntityVersion);
```

X.2) defining related object graph bounds

refers to **B1)**

When inspection of the graph of referencing objects is needed it is neccesary to define the bounds of the graph to be walked. For example a References is modified a possible branch of referencing objects graph is Reference --> nomenclaturalReference.Name --> name.Taxon --> taxon.TaxonNode --> children.TaxonNode ..... At the point where the walk along the graph reaches the classification tree, that is a TaxonNode it should stop.

Ways to define the bounds:

- Stop classes: TaxonNode, FeatureTreeNode ... (All node types in hierachies?)
- ReferencingObjectScopes: This would be a project specific scope definition. For Phycobank the scope would encompass all nomenclaatural acts (Names, TypeDesignations) which belong to the same Registration.
- ... other ?

But what if the name of a Genus is being changed? we can't stop in this case, all taxonomic children would need to be taken into account.

... To be continued ....

X.3) unnoticed accidentally changing of second level referencing objects

refers to **C)**

A user changing an author may be aware of the fact that the reference in which this Person is being used will be modified. But the reference may be used in further contexts. This could lead to unnoticed modifications.

The here described potential risk especially occurs in situations where the user is notified about the first level referencing objects that will be affected by the change. This conveys a security to the user which is not really given. Therefore solutions relying on the users decision are not appropriate.

Related issues:

Related to EDIT - feature request #3709: [E+M][Editor] sufficient rights mana...	**Resolved**
Related to EDIT - bug #6886: Entity creation for users having only CREATE may...	**Duplicate**
Related to PhycoBank - bug #6185: prevent from erroneous author or reference ...	**Closed**
Copied to EDIT - feature request #6867: explicitely assign and revoke UPDATE...	**Closed**

History

#1 - 08/18/2014 12:15 PM - Andreas Kohlbecker

- Keywords changed from permission security to permission,security,Euro+Med,Migration

#2 - 02/08/2017 12:04 AM - Andreas Müller

- Description updated

#3 - 06/20/2017 08:18 AM - Andreas Kohlbecker

- Related to feature request #3709: [E+M][Editor] sufficient rights management for E+M workflow added

#4 - 07/24/2017 10:28 AM - Andreas Kohlbecker

- Tags changed from permission, security, euro+med, migration to permission, security, euro+med, migration, phycobank

- Private changed from Yes to No

#5 - 07/25/2017 12:45 PM - Andreas Kohlbecker

- Status changed from New to In Progress

- Priority changed from Priority14 to Highest

- % Done changed from 0 to 10

- Severity changed from critical to blocker

#6 - 07/25/2017 01:07 PM - Andreas Kohlbecker

- Description updated

#7 - 07/25/2017 01:08 PM - Andreas Kohlbecker

- Status changed from In Progress to Feedback

I think solution **A)** as described above is the best idea so far we had.

What do you think Andreas M?

#8 - 07/25/2017 01:12 PM - Andreas Kohlbecker

- Description updated

#9 - 07/25/2017 01:12 PM - Andreas Kohlbecker

- Description updated

#10 - 07/25/2017 01:12 PM - Andreas Kohlbecker

- Description updated

#11 - 07/26/2017 10:53 AM - Andreas Müller

- Description updated

#12 - 07/26/2017 03:04 PM - Andreas Kohlbecker

- Description updated

After sleeping one night over it, i came to the following conclusions:

- A) This approach is not appropriate in the context of applications like phycobank where entity must stay editable as long as the registration process is not completed, this can take several weeks.
- B1) The only really reliable strategy and maybe not so extensive in terms of computation as initially assumed.
- B2) Elegant and simple, but it might cause problems and conflicts in phycobank. An example for problems that would arise: A registration for a publication with multiple new names whereas the authors of the name differ from the publication authors. The name X has the authors A and B whereas the article is published by the authors A, B and C. A and B have two referencing objects in this case and are blocked.

#13 - 07/26/2017 03:05 PM - Andreas Kohlbecker

- Description updated

#14 - 07/26/2017 05:24 PM - Andreas Kohlbecker

- Description updated

#15 - 07/26/2017 05:58 PM - Andreas Kohlbecker

- Description updated

#16 - 07/26/2017 06:10 PM - Andreas Kohlbecker

- Description updated

#17 - 07/26/2017 06:23 PM - Andreas Kohlbecker

- Description updated

#18 - 07/27/2017 11:00 AM - Andreas Kohlbecker

- Description updated

#19 - 07/27/2017 11:18 AM - Andreas Kohlbecker

- Description updated

#20 - 07/27/2017 11:20 AM - Andreas Kohlbecker

- Description updated

#21 - 08/01/2017 01:10 PM - Andreas Kohlbecker

- Description updated

#22 - 08/01/2017 01:15 PM - Andreas Kohlbecker

- Description updated

#23 - 08/01/2017 01:29 PM - Andreas Kohlbecker

After an in depth discussion we decided that for phyconbank the strategy **D)** (per instance UPDATE & DELETE permission) would be the most appropriate:

- a submitter will the per instance UPDATE+DELETE permission when he creates a Reference, TeamOrPersonBase, Name instance.
- once a registration is set to the states rejected, ready or published the UPDATE+DELETE permission must be revoked again, so that the registered name and references are protected from being changed after the editing registration workflow has ended.

The RegistrationStateManager (#6655) could therefore be responsible for revoking authorities. Assignment of authorities however should not be managed in this state machine.

#24 - 08/01/2017 01:38 PM - Andreas Kohlbecker

- Copied to feature request #6867: explicitely assign and revoke UPDATE & DELETE permission per enitity in the registration workflow added

#25 - 08/04/2017 01:00 PM - Andreas Kohlbecker

- Related to bug #6886: Entity creation for users having only CREATE may fail in long running conversations added

#26 - 09/12/2017 09:52 AM - Andreas Kohlbecker

- Related to bug #6185: prevent from erroneous author or reference changes added

#27 - 09/22/2017 04:11 PM - Andreas Kohlbecker

- Target version changed from Euro+Med Migration to Release 4.11

- % Done changed from 10 to 40

#28 - 10/19/2017 12:22 PM - Andreas Kohlbecker

- Status changed from Feedback to In Progress

#29 - 11/06/2017 12:33 PM - Andreas Kohlbecker

- Priority changed from Highest to Priority13

- Target version changed from Release 4.11 to Reviewed Next Major Release

- Severity changed from blocker to normal

The very important task is now being managed in #6867 this issue here therefore is no longer really pressing since it merely only contains all the thoughts on the strategies that have been discussed. Therefore I am lowering the severity and priority and I am moving the ticket to the reviewed ones. But what about the status. It is no longer in progress but also any other status doesn't fit for a such ticket on hold.

#30 - 02/19/2019 03:34 PM - Andreas Müller

- Priority changed from Priority13 to Priority10