Morpho - Bug #5737

Refactor Access page to use SystemMetadata rather than EML

11/19/2012 03:37 PM - ben leinfelder

Status:	Resolved	**Start date:**	11/19/2012
Priority:	Normal	**Due date:**	
Assignee:	Jing Tao	**% Done:**	0%
Category:	morpho - general	**Estimated time:**	0.00 hour
Target version:	2.0.0	**Spent time:**	0.00 hour
Bugzilla-Id:	5737		

Description

Or a combination of the two different ways of defining access control rules. Still TBD

Related issues:

Blocked by Morpho - Bug #5736: Morpho DataONE refactor - tracker	**New**	**11/19/2012**

History

#1 - 12/10/2012 03:57 PM - ben leinfelder

I'm working on moving EML access parsing from Metacat into the utilities project so that it can be shared by Morpho.
For cases where there is no pre-existing SYystemMetadata, we'd use the EML-based access rules to prime the SM then use the SM from then on.
This represents a bit of refactoring in both Metacat, utilities and Morpho, but I think it is worth it so that we don't end up with yet another EML parser in Morpho.

#2 - 12/11/2012 04:11 PM - ben leinfelder

My current approach is to generate SM.AccessPolicy from the EML that we have (including after it is modified in the UI but before being saved). This is working well so far for cases when the EML and the SystemMetadata are in synch (and it pretty much keeps them in synch since that is the only way in the UI to edit the AccessPolicy) . But if we encounter an EML file on the network that does not express the same AccessPolicy in the SystemMetadata there will be confusion.

#3 - 01/09/2013 11:13 AM - Jing Tao

Now the access page will read and store the access information from the SystemMetadata object in both the data package and entity leve. Also, any access rules modification will change the eml file as well.

However, there is an issue in the entity access. If a user modifies the access rule for an entity, the information will be stored in both the EML and the SystemMetadata object associated with the entity. When the user saves the data package, the eml file and the SystemMetadata object associated with the eml will be saved; the entity and the SystemMetadata object associated the entity will be not since the entity is not dirty. So the new access rule will be lost because the new SystemMetadata object hasn't been serialized. I propose to have a flag in the Enity class - systeMetadataIsDirty. If the flag is true, the SystemMetadata object associated with the entity will be serialized but the Entity file will not.

#4 - 01/09/2013 11:22 AM - ben leinfelder

Saving the SM for entities when the access policy has changed (but the actual data file has not) is fine for local save. Unfortunately, there is no MN.saveSystemMetadata() method and all modifications to SystemMetadata on a server are intended to be made on the CN, not the MN. There are, of course, some issues with this decision since we don't know when the saved object will ever be synchronized to the CN and we may even be operating in cases where the MN is never a member of any federation.

We may need to discuss this CN/MN architecture decision more, but in the meantime I think it will be fine to add an isSystemMetadataDirty flag for each data entity.

#5 - 01/09/2013 11:52 AM - Jing Tao

Maybe we need to add the method - MN.saveSystemMetadata().

Actually i want to add the flag - isSystemMetadataDirty in the D1Object level because the SystemMetadata class field is in the D1Object class rather than the Entity class.

#6 - 01/09/2013 11:57 AM - ben leinfelder

adding D1Object.isSystemMetadataDirty might be tricky since it is in the d1 library

#7 - 01/09/2013 04:44 PM - Jing Tao

Ben and I discussed the issue about how to save the the entity only with the system metadata modification. We came up with a temporary solution - even though there is only system metadata change, we still label the entity dirty. This solution works but has a drawback - there data file was

duplicated.

We will open another bug to address the issue.

#8 - 03/27/2013 02:31 PM - Redmine Admin

Original Bugzilla ID was 5737