Infrastructure - Bug #4303

Fix potential bug where an object could be created and accepted for replication simultaneously.

2014-03-06 22:42 - Roger Dahl

Status:	New	**Start date:**	2014-03-06
Priority:	Low	**Due date:**	2015-01-05
Assignee:	Roger Dahl	**% Done:**	0%
Category:	d1_mn_GMN	**Estimated time:**	0.00 hour
Target version:	Release Backlog		
Milestone:	None	**Story Points:**	
Product Version:	*		

Description

Use SQL "select for update" (or similar) to lock the replication queue while a regular create takes place, so that a replication request cannot be accepted for an object at the same time as that object is being created. GMN already checks, in create(), that a replication request does not exist for the object. It also checks for the opposite, but there is a tiny window in which both could be created at the same time.

History

#1 - 2014-08-20 01:47 - Roger Dahl

This ticket is about improving handling for the very unlikely scenario of a CN calling MNReplication.replicate() with a given PID at the same time as a user calls MNStorage.create() for that same PID. The basic problem is that there are two separate tables holding the existing objects and the queue of replication requests and a given PID should never be in both tables at once. However, it's not possible to specify a unique constraint that covers two tables. GMN already checks, when processing a create(), that the PID is not already in the replication queue and it checks, when processing replicate(), that the PID is not already in the object table. The potential problem happens if the calls are processed concurrently, so that each call finds the PID not to exist in the opposite table and then goes ahead and inserts it.

The initial idea I had of using "select for update" is not going to work because "select for update" does not lock non-existing rows. Other ideas for resolving this revolve around creating a separate table for the value that should be unique (the PIDs in this table) and then using foreign key relationships:

http://stackoverflow.com/questions/10337944/sql-unique-constraint-across-multiple-tables

As things are currently implemented, if the same PID does end up getting inserted into both tables, a failure will be detected when the async replication processing happens. At that time, the entry will be marked as failed but it will keep getting retried until someone manually removes it from the queue.

#2 - 2014-10-01 20:48 - Robert Waltz

- Assignee changed from Roger Dahl to Mark Servilla

#3 - 2014-10-01 20:50 - Robert Waltz

- Parent task deleted (#4299)

#4 - 2014-10-01 20:50 - Robert Waltz

- translation missing: en.field_remaining_hours set to 0.0

- Tracker changed from Task to Bug

- Due date set to 2014-10-01

- Target version set to Release Backlog

#5 - 2015-01-05 18:43 - Mark Servilla

- Due date changed from 2014-10-01 to 2015-01-05

- Assignee changed from Mark Servilla to Mark Flynn

#6 - 2016-04-21 22:38 - Roger Dahl

- Assignee changed from Mark Flynn to Roger Dahl