Grouper USDU v2.5+

The new USDU acronym in v2.5.x: Universal Subject Daemon Utility. Yes we renamed it so we don't need to have people change their configs. 🙂

In a Grouper v2.5 build, there is the USDU daemon that resolves subjects (in batch) and does some tasks

1. Resolves them, and if not resolved, then call USDU logic to start the unresolvable process (i.e. they can be unresolvable for X days until removed from Grouper)
2. Compares their search and sort strings with whats in the grouper_members table and batch update if there are changes
3. If a provisioner is using a subject scriptlet to handle subjects during provisioning, calculates the values and compare to the sync tables and bulk update if there are changes
    a. Note, we need to get all the changes needed, and block any provisioners from running so we dont step on toes, do the changes, and then unblock

Provisioning configuration about subjects

In syncing data with target systems, the link to an entity might not be the same sourceId/subjectId tuple that Grouper uses. Maybe an attribute or expression needs to be evaluated against the subject to link the entities. This places an undo burden on these operations since resolving subjects from an external system is expensive performance-wise. Further there are issues when subjects become unresolvable to clean up provisioned data where the links are broken.

You might have a DN in an ldap provisioner that is based on a subject attribute. We want to keep a copy of that in the Grouper database so we can efficiently do provisioning. If the attribute changes throughout the day, there will be an error and the subject will be re-resolved as needed.

If there is a config to cache this, then it will be cached by the new USDU.

There are up to four values that can be cached for subjects in the grouper_sync_member table. Different provisioners will use these however they want. It is intended that the "from" ID's will be Grouper or Subject API things, and the "to" ID's will be for data in the provisioning target (e.g. an attribute of the person in LDAP or a uuid from box).

grouper-loader.properties (note this will be configured on the UI in a wizard). If the provisioner is enabled, and there is configuration in member(From|To)Id(2|3), and the "auto" setting is not there or blank or true, then do the update in the daemon.

```
if provisioner.<configName>.common.enabled is false, then skip

provisioner.<configName>.common.subjectLink.memberFromId2 = ${subject.whatever}
provisioner.<configName>.common.subjectLink.autoMemberFromId2 = true (blank defaults to true)

provisioner.<configName>.common.subjectLink.memberFromId3 = ${subject.whatever}
provisioner.<configName>.common.subjectLink.autoMemberFromId3 = true (blank defaults to true)

provisioner.<configName>.common.subjectLink.memberToId2 = ${subject.whatever}
provisioner.<configName>.common.subjectLink.autoMemberToId2 = true (blank defaults to true)

provisioner.<configName>.common.subjectLink.memberToId3 = ${subject.whatever}
provisioner.<configName>.common.subjectLink.autoMemberToId3 = true (blank defaults to true)
```

Lock on a provisioner

Get all sync members for a provisioner, change them, and persist back

```
GcGrouperSync gcGrouperSync = GcGrouperSyncDao.retrieveOrCreateByProvisionerName(null, <configName>);
GcGrouperSyncJob gcGrouperSyncJob = gcGrouperSync().getGcGrouperSyncJobDao().jobRetrieveOrCreateBySyncType
("usduSubjectCacheUpdater");
gcGrouperSyncJob.waitForRelatedJobsToFinishThenRun(true);

try {
// thread to update heartbeat

        this.setJobState(GcGrouperSyncJobState.running);
        this.setHeartbeat(new Timestamp(System.currentTimeMillis()));
        this.grouperSync.getGcGrouperSyncJobDao().internal_jobStore(this);



  List<GcGrouperSyncMember> gcGrouperSyncMembers = this.gcTableSync.getGcGrouperSync().
getGcGrouperSyncMemberDao().memberRetrieveAll();

  ...make changes...

  # this will persist in batch only objects that have changed since being retrieved
  int objectsStored = gcTableSync.getGcGrouperSync().getGcGrouperSyncDao().storeAllObjects();
} finally {
  GcGrouperSyncHeartbeat.endAndWaitForThread(this.gcGrouperSyncHeartbeat);

  gcGrouperSyncJob.assignHeartbeatAndEndJob();
}
```

See Also

Subject Change Daemon